MASTER'S THESIS

Real-time Blind Separation and Deconvolution of Real-world signals

by Yu Mao
Advisor: P.S. Krishnaprasad

CAAR MS 2003-1
(ISR MS 2003-5)

Center for Auditory
and Acoustic Research

The Center for Auditory and Acoustic Research (CAAR) is a consortium of researchers from six universities working in partnership with Department of Defense laboratories and industry. CAAR is funded by the Office of Naval Research through a 1997 Department of Research Initiative.

Web site http://www.isr.umd.edu/CAAR/

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **2003**	2. REPORT TYPE	3. DATES COVERED **00-00-2003 to 00-00-2003**

4. TITLE AND SUBTITLE **Real-time Blind Separation and Deconvolution of Real-world signals**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Maryland, College Park,Center for Auditory and Acoustic Research (CAAR),College Park,MD,20742**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
We present a reallistic and robust implementation of Blind Source Separation and Blind Deconvolution. The algorithm is developed from the idea of natraul gradient learning, wavlet filtering and denoising, and the characteristic of different sound source. Several hardware pieciecs are integrated, including a mobile robot NT workstation and DSP chip to achieve the real time separation of real world signal. Besides, a method of judging the separation performance without knowing the mixing matrix (mixing filter) is proposed and verified.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**81**	

ABSTRACT

Title of Thesis:	REAL-TIME BLIND SEPARATION AND DECONVOLUTION OF REAL-WORLD SIGNALS
Degree candidate:	Yu Mao
Degree and year:	Master of Science, 2002
Thesis directed by:	Professor P. S. Krishnaprasad Department of Electrical and Computer Engineering

We present a reallistic and robust implementation of Blind Source Separation and Blind Deconvolution. The algorithm is developed from the idea of natraul gradient learning, wavlet filtering and denoising, and the characteristic of different sound source. Several hardware pieciecs are integrated, including a mobile robot, NT workstation and DSP chip to achieve the real time separation of real world signal. Besides, a method of judging the separation performance without knowing the mixing matrix (mixing filter) is proposed and verified.

REAL-TIME BLIND SEPARATION AND DECONVOLUTION OF REAL-WORLD SIGNALS

by

Yu Mao

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2002

Advisory Committee:

Professor P. S. Krishnaprasad, Chairman
Professor Shihab Shamma
Professor Carlos Berenstein

©Copyright by

Yu Mao

2002

DEDICATION

To my parents Zaisha Mao, Junxian Zhou and my husband Fumin Zhang for their encouragement and support throughout my life.

ACKNOWLEDGEMENTS

I would to thank my advisor, Dr. P.S. Krishanprasad for all his guidance and encouragement. He consistently challenged me to do my best. I would also thank Dr. Shihab Shamma and Dr. Carlos Berenstein for being my committee members. Everybody in the ISL lab has contributed his or her part in this robot project. I need to thank them all, especially Sean Anderson and Fumin Zhang for the motion control and sound card programming, and Cliff Knoll for providing the communication utility between the robot and python board.

TABLE OF CONTENTS

List of Tables vi

List of Figures vii

1 Introduction 1
- 1.1 Background 1
- 1.2 Overview 2

2 Analysis and Comparsion of BSS/BSD Algorithms 4
- 2.1 Preprocessing 5
  - 2.1.1 Whitening 5
  - 2.1.2 Filtering and de-noising 6
- 2.2 Error functions 7
  - 2.2.1 Cumulant 7
  - 2.2.2 Kurtosis 8
  - 2.2.3 Kullback-Leibler divergence 8
  - 2.2.4 Maximum likelihood 9
- 2.3 Learning algorithms 10
  - 2.3.1 Stochastic gradient method 10
  - 2.3.2 Natural gradient method 10
  - 2.3.3 Non-holonomic learning 12
  - 2.3.4 EM algorithm 15
  - 2.3.5 Stability analysis 16
- 2.4 Variation in modeling 18
  - 2.4.1 How sound spread in closed space 19
  - 2.4.2 Filter convolution model 19
  - 2.4.3 State space model 20
- 2.5 Blind deconvolution 21
  - 2.5.1 Learning under state-space model 22
  - 2.5.2 Learning under the filter deconvolution model 25
  - 2.5.3 Kuicnet approach 28

3 Sub-band Based ICA Algorithm **29**
3.1 Filter-bank structure in the human ear 30
3.2 Sub-band ICA 30
3.3 Implementation of the algorithm 34
3.3.1 Use adaptive basis selection in the wavelet packet 34
3.3.2 Selecting the bands to perform ICA 34
3.3.3 Introducing the convolution model into the sub-band algorithm 35

4 Working with Real World Signal **36**
4.1 Hardware and environment 36
4.1.1 Styrofoam head and robot 38
4.1.2 Windows NT workstation 38
4.1.3 DSP processor 39
4.1.4 Python board 40
4.1.5 Room environment 40
4.2 Considerations for a real time implementation 40
4.2.1 Data communication 41
4.2.2 Optimizing the code for speed and memory size 41
4.2.3 Tuning parameters for the best performance 42
4.2.4 Smoothing between blocks 43
4.3 Performance evaluation criterion 43

5 Experimental Results **46**
5.1 The effect of source type on performance 46
5.2 The effect of source distance and angle 49
5.3 The effect of room acoustics 54
5.4 Over-complete and under-complete mixtures 54
5.5 Conclusion and future work 56

A Software Structure **59**
A.1 Connection between Coreco board and NT Host 59
A.2 Sub-band ICA algorithm 61
A.2.1 Data structure 61
A.2.2 Algorithm 65
A.3 File Structure 65

B Project Directory **67**

Bibliography **70**

LIST OF TABLES

4.1 Comparison of two kind of performance index 45

5.1 Simulation result of sub-band ICA algorithm on different type of sources 47
5.2 Simulation result of real-time separation of real-world recording on different type of sources 48
5.3 Performance index of sub-band ICA on real-world recording on different source angle 52
5.4 Performance index of sub-band deconvolution on real-world recording on different source angle 54

LIST OF FIGURES

3.1 The model of human ear 31
3.2 The block diagram of Subband ICA algorithm 33

4.1 The block diagram of the hardware setting for this project 37

5.1 The separation of two man's voice 47
5.2 The separation of voice and music 48
5.3 The layout of Experiment 49
5.4 Comparing the instantaneous mixing model and the real-world mixing model 50
5.5 Separation of real-world mixture in in position A 51
5.6 Separation of real-world mixture in position B 52
5.7 Result of deconvolution algorithm in position A and B 53
5.8 Impulse response of demixing filter 55

A.1 Host program and server program 60
A.2 Circular buffer and wavelet filter bank 64
A.3 Master DSP program and slave DSP program 66

Chapter 1

Introduction

1.1 Background

In many application, there is more than one signal source in a system and the system introduces noise and different time delays from the signal sources to the observer(or sensor). It is desirable to separate mixtures and recover the original data as closely as possible. In most cases, the original source and the system transfer function are unknown. Such a separation problem is called Blind Source Separation/Deconvolution (BSS/BSD), depending on whether the delay effect is considered in the model or not.

The BSS(BSD) problem has been receiving increasing attention in the last decade because of applications in speech enhancement and recognition, biomedical signal analysis, image processing and telecommunication.

In the work I will discuss here the major concern is with the case of natural sound sources mixed in the process of air transmission in a closed space and received by a group of microphones in the same space. Different models and separation/deconvolution algorithms are analyzed and compared in such an environment and a new algorithm that makes use of the statistical distribution characteristics of natural sound combined with a frequency domain analysis is used to achieve better results in recovering original data in real time. Many open questions, including extension to time varying systems, changing number of sources, noise reduction, signal extraction and varying receiver location, are addressed to achieve a practical and robust solution.

1.2 Overview

In an environment with n sound sources, m sensors are placed. The goal is to separate sound from different sources and recover the original signals. We represent the sources as $s_1(t), s_2(t), ..., s_n(t)$. Since the sound sources are physically independent of each other, we assume the signals are mutually statistically independent. The original signals $s_1(t), s_2(t), .., s_n(t)$ are unobservable. The m outputs from sensors are denoted by $x_1(t), x_2(t), ...x_m(t)$. These also intensity functions of time and they are linear combination of sources.

The model can be written as $x_i(t) = \sum_{j=1}^{n} s_j(t)a_{ij} + n_i(t), i = 1, 2, ..., m$, where $n_i(t)$ is observation noise. The matrix $\{a_{ij}\}$ represents the mixing in the environment. Since the environment may be changing, $\{a_{ij}\}$ in general is a function of time. Let $x = [x_1(t), x_2(t), ..., x_m(t)]^T$ be the observation data vector, $s = [s_1(t), s_2(t), ..., s_n(t)]^T$ the source vector and $A = \{a_{ij}\}$ be a time-varying matrix. We have the vector form

$$x = As \tag{1.1}$$

A related task is blind deconvolution where we suppose the mixing channel has delay. In this model the observation $x(k)$ is assumed to be produced from $s(k)$ in the following manner

$$x(k) = \sum_{p=-\infty}^{\infty} H_p s(k-p) \tag{1.2}$$

Where each H_p is an $n \times m$ matrix of unknown mixing coefficients at lag p. The goal is to calculate $y(k) = [y_1(k) \cdots y_m(k)]^T$ of possibly scaled and delayed estimates of the source signals in $s(k)$ from $x(k)$ using a causal FIR filter given by

$$y(k) = \sum_{p=0}^{L} W_p(k) x(k-p) \tag{1.3}$$

where $W_p(k), 0 \leq p \leq L$ is a $(m \times n)$ matrix satisfy

$$W(z)H(z) = P\Lambda(z)$$

where P is a permutation matrix and $\Lambda(z)$ is a diagonal matrix with $\lambda_i z^{-\tau_i}$ as the diagonal entries.

There are several important assumptions about this BSS/BSD model:

1. The source signals are mutually statistically independent for each sample.
2. At most one of the source signals has a Gaussian distribution.
3. Each source signal is a stationary stochastic process.
4. Besides being stationary, the signals are all ergodic so that time averages can be used to estimate statistical distributions.

In Chapter 2 different ways to solve the BSS/BSD problem will be analyzed. In Chapter 3 an improved subband-based Independent Component Analysis (ICA) algorithm will be discussed. In Chapter 4 we describe the real world environment and provide details on implementation of a real-time ICA algorithm for real-world signals. In Chapter 5 the test results will be discussed and some conclusions are drawn.

Chapter 2

Analysis and Comparsion of BSS/BSD Algorithms

The essential goal of BSS/BSD problem is to recover unknown source signals from mixed observations.

In this chapter the instantaneous mixing model described below will be used unless stated otherwise:

$$x = As + n \tag{2.1}$$

where s is the vector of original signals, x is the observed signals, A is the unknown mixing matrix, a simplified representation of environment, and n is the observation noise.

The corresponding separated system should be:

$$y = Wx \tag{2.2}$$

where W is the separation matrix and y is the recovered signal.

2.1 Preprocessing

Before applying the ICA algorithm on the data some preprocessing of the data can help to reduce the noise, accelerate the convergence speed and reduce computation. The typical ways are:

2.1.1 Whitening

Before ICA, the standard Principle Component Analysis (PCA) algorithm can be applied on the data. PCA produces uncorrelated signals. This is closer to the final goal of ICA, independent signals, as compared to the raw source data.

Denote the covariance matrix of sources as

$$R_{ss} = Es(t)s^T(t) \tag{2.3}$$

where R_{ss} is a positive definite diagonal matrix. The covariance matrix of the observed signal is

$$R_{xx} = Ex(t)x^T(t) = HR_{ss}H^T \tag{2.4}$$

PCA searches for an orthogonal matrix Q such that the components of

$$y(t) = Qx(t)$$

are uncorrelated. In other words the covariance matrix of the output y is diagonal.

$$R_{yy} = E[y(t)y^T(t)] = QR_{xx}Q^T = V \tag{2.5}$$

In the case where the original signals are Gaussian, PCA results in independent output. In general further computation is required on y to obtain independent output components.

Let

$$\widetilde{x}(t) = V^{-1/2}Q^T x(t) \tag{2.6}$$

such that

$$E\widetilde{x}(t)\widetilde{x}^T(t) = I \tag{2.7}$$

Thus

$$y = W\widetilde{x}(t) \tag{2.8}$$

extracts the independent signals, where W is the separating matrix.

In the work of Koehler, Lee and Orglmeister [15], the mixed signal is further decorrelated by fourth order statistics before the ICA algorithm is applied. Let

$$W_4 = (E[||\widetilde{x}||^2\widetilde{x}\widetilde{x}^T])^{-1/2} \tag{2.9}$$

and

$$\widehat{x} = W_4\widetilde{x} \tag{2.10}$$

is the input for core ICA algorithm.

2.1.2 Filtering and de-noising

If the frequency range of signals is known a band pass filter can be used to remove all the noise with spectrum outside of that frequency range. While if noise is narrow-band a band stop filter can be used to remove it.

For white noise, thresholding combined with a wavelet method is effective. The method proposed by David Donoho [10], removes Gaussian noise and produces an estimation that is balanced between the two goals of smoothness and small bias.

This method produces very good results in the presence of noise and will be further discussed in Chapter 3.

2.2 Error functions

Since the signals from different sources are assumed to be independent, the goal of the BSS problem is equivalent to achieving independent output by multiplying the observed signals with an invertible matrix or applying a filter on the observation. However the definition of independence

$$P(x_1, x_2, ..., x_n) = P(x_1).P(x_2)...P(x_n) \tag{2.11}$$

is not directly usable in an error function because the statistical distribution of the original signal is unknown and cannot be estimated from the mixed signal. Different approaches have been thus suggested to achieve an implementable error function as below.

2.2.1 Cumulant

Jutten and Herault, inspired by neurobiological research, first proposed this heuristic learning algorithm to solve the BSS problem using the cumulant as the loss function[14].

In contrast with PCA, which decorrelates signals based on the covariance matrix, in ICA higher order statistics are used to find the independent components. The error function for PCA is:

$$L(W) = y_m^2(k) \tag{2.12}$$

The convergence of $E(\frac{dw_{ik}}{dt}) = 2q_{mi}E(y_i(t,f)y_k(t,f)) = 0$ means $y_i(t,f)$ and $y_k(t,f)$ are uncorrelated, as we can see from section 2.1.1 about prewhitening. To achieve the independence between $y_i(t,f)$ and $y_k(t,f)$, we use two different nonlinear and odd functions

$$\frac{dW_{ik}}{dt} = af(y_i(t,f))g(y_k(t,f))$$

$\tanh(y)$ and y^3 are the commonly used non-linear functions.

2.2.2 Kurtosis

Kurtosis is a measure of how far a statistical distribution is away from a Gaussian distribution. According to the Central Limit Theorem, the sum of n independent identically distributed random variables has a Gaussian distribution as n goes to infinity. For n independent but not identically distributed random variables, if their PMF's satisfy certain conditions, then the central limit theorem still holds.

A Gaussian random variable has kurtosis equal to 0. The closer kurtosis is to 0, the closer the random variable is to a Gaussian random variable. Kurtosis can then be used as a cost function for separation and deconvolution. It is used more effectively for deconvolution and will be discussed later in Section 2.5.

2.2.3 Kullback-Leibler divergence

The Kullback-Leibler divergence (also called relative entropy) between two statistical distributions is defined as

$$D(p \parallel q) = \int p(x) \log(p(x)/q(x)) dx$$

Although it is not symmetric and does not satisfy the triangle property, Kullback-Leibler divergence is still considered a measure of “distance” between two statistical distributions.

To achieve independent output means to minimize the K-L divergence between the joint distributions of the estimated sources y and the product of the marginal distributions of the y_i

$$D_{f_y \| \widetilde{f}_y} = \int_{-\infty}^{\infty} f_y(y) \log(\frac{f_y(y)}{\prod_{i=1}^{n} \widetilde{f}_{y_i}(y_i)}) = -H(y) + \sum_{i=1}^{n} \widetilde{h}(y_i)) \tag{2.13}$$

where $H(.)$ is entropy of the joint probability distribution.

It can be shown that the above error function is consistent with the following [9]:

$$I(y,x) = H(y) - H(y/x) \tag{2.14}$$

where $I(y,x)$ is the mutual information in output y about input x, $H(y)$ is the entropy of y, $H(y/x)$ is the entropy of output y that did not come from input x.

We thus have

$$\frac{\partial}{\partial W} I(y,x) = \frac{\partial H(y)}{\partial W} \tag{2.15}$$

since $H(y/x)$ does not depend on W.

2.2.4 Maximum likelihood

To estimate one set of data from another set, the likelihood function is highly useful. It is defined as a product of factors obtained by marginalizing over the latent variable. In this case

$$p(x|A) = \prod_{i=1}^{n} p_i(x_i|A) \tag{2.16}$$

In our goal is to achieve $W = A^{-1}$

$$p(y|x, W^{-1}) = \prod_{i=1}^{n} p_i(y_i|x, W^{-1}) \tag{2.17}$$

From this we take the negative log likelihood as error function

$$l(y,W) = -\log|det(W)| - \sum_{i=1}^{n} \log p_i(y_i) \tag{2.18}$$

$$L(W) = E[l(y, W)] \tag{2.19}$$

Here $W \in Gl(n)$, the matrix Lie group of $n \times n$ nonsingular matrices.

Remark: In [16] David Mackay proved this error function is consistent with K-L distance.

2.3 Learning algorithms

2.3.1 Stochastic gradient method

For the cost function $L(W)$, the algorithm derived from steepest descent method is:

$$\frac{dW}{dt} = \frac{dL(W)}{dW} \tag{2.20}$$

The error function $L(W)$ usually contains the expectation operation which can not be implemented in the process of learning. Since the original signal is considered to be ergodic, time average can be used to replace the expectation.

2.3.2 Natural gradient method

The natural gradient algorithm was proposed by Shunichi Amari in 1995. In [1] Amari proved this algorithm possesses the equivariance property and achieves the Fisher Efficiency.

An ideal property for a learning algorithm is covariance or equicovariance, which means the algorithm should give the same results independent of the units in which quantities are measured. The steepest descent rule does not give a covariant algorithm, because the two sides of the equation are not consistent in dimension.

Suppose

$$\Delta w_i = \mu \frac{\partial L}{\partial w_i} \tag{2.21}$$

Then choose μ of a particular dimension will only result in a covariant algorithm if all the w_i have the same dimension. However if we take

$$\Delta w_i = \mu \sum_j M_{ij} \frac{\partial L}{\partial w_i} \tag{2.22}$$

where M is a matrix whose (i, j)th element has dimension $[w_i w_j]$, then the algorithm is covariant. There are two ways to get such matrices, namely metric method and curvature method.

Newton's algorithm is an example of getting M from curvature. In this algorithm we have

$A = -\nabla\nabla L$ and $M = A^{-1}$

In the natural gradient algorithm we get the matrix M from metrics.

The steepest descent direction of a function in a Riemannian space is given by

$$-\widetilde{\nabla} L(w) = -G^{-1}(w)\nabla L(w) \tag{2.23}$$

where G is the Riemannian metric and $\nabla L(w)$ is the standard gradient. Using equation 2.18 and 2.19, by taking derivative on both sides we have:

$$\begin{aligned} dl &= -d(\log|det(W)|) - \sum_{i=1}^{n} d(\log p_i(y_i)) &(2.24)\\ &= -tr(dWW^{-1}) + \phi(y)^T dy &(2.25)\end{aligned}$$

where

$$\phi_i(y_i) = -\frac{d}{dy_i}\log(p_i(y_i))$$

The Riemannian metric of a statistical model is defined by the Fisher information matrix [9]. That is.

$$g_{ij}(w) = E\{\frac{\partial \log p(x,w)}{\partial w_i}\frac{\partial \log p(x,w)}{\partial w_j}\} \tag{2.26}$$

Then

$$dX = dWW^{-1} \tag{2.27}$$

is to be viewed as an element of the cotangent space to $Gl(n)$ at the identity I. Define the co-metric

$$\langle dX, dX \rangle_I = tr(dXdX^T) \tag{2.28}$$

then

$$\langle dW, dW \rangle_W = tr((dWW^{-1})(dWW^{-1})^T) \tag{2.29}$$

With respect to this metric we get the gradient

$$\Delta W = [I - \phi(y(t))y^T(t)]W \tag{2.30}$$

The natural gradient algorithm is given by the following :

$$W_{t+1} = W_t - \eta(t)F\{y(t), W_t\} \tag{2.31}$$

where

$$F\{y(t), W_t\} = [I - \phi(y(t))y^T(t)]W \tag{2.32}$$

2.3.3 Non-holonomic learning

In the last section we derived the natural gradient algorithm. However, this learning algorithm can not distinguish W with ΛW, where Λ is a diagonal matrix with nonzero diagonal entries. Thus given W, the equivalence class $S_W = \{W' | W' = \Lambda W\}$ is a n-dimensional subspace of S, where $S = \{W\} = Gl(n)$. Shunichi Amari, T.P. Chen and A. Chichocki found a way to improve learning performance by adding non-honolomic constraints to the natural gradient algorithm in [3].

Three kinds of possible constraint may be used to restrict the search direction of the above algorithm.

1. hard restriction

$$f_i(W) = 0, \quad i = 1, ..., n$$

   with the typical and most simple choice

$$f_i(W) = W_{ii} - 1$$

2. soft restriction

$$E(f_i(y_i)) = 0, \quad i = 1, ..., n$$

   The typical choice is

$$f_i(y_i) = y_i^2 - 1$$

   which guarantees the variation of the recovered signals are 1.

3. non-holonomic constraint

   Define $\Delta X_t = \Delta W_t W_t^{-1}$. Then the constraints are given by

$$\Delta X_{ii} = 0, i = 1, ...n$$

   This constraint implies that the trajectories of the dynamics are always orthogonal to S_W while not being restricted to a fixed sub-manifold.

Any matrix can be uniquely represented by the sum of two matrices, a diagonal matrix and a matrix with all diagonal elements zero.

Thus we can write

$$Gl(n) = \mathcal{A} + \mathcal{B} \tag{2.33}$$

where

$$\mathcal{A} = \{A \in \mathcal{R}^{n\times n} | A = diag\{a_1, a_2, ..., a_n\}\}$$

$$\mathcal{B} = \{B \in \mathcal{R}^{n\times n} | b_{11} = b_{22} = ... = b_{nn} = 0\}$$

Claim

$$[\mathcal{B}, \mathcal{B}] = sl(n) \tag{2.34}$$

where

$$[\mathcal{B}, \mathcal{B}] = \{x \in \mathcal{B}, y \in \mathcal{B}, [x, y] = xy - yx\}$$

Proof:

Suppose $Z \in sl(n)$ i.e. $\sum_{i=1}^{n-1} z_{ii} = z_{nn}$.

Denote I_{ij} as a $n \times n$ matrix with the (i, j)th element as 1 and other elements as 0. Denote J_i as a $n \times n$ matrix with the (i, i)th element as 1 and other element as 0. Then

$$I_{ij} \in \mathcal{A}, \quad J_i \in \mathcal{B} \tag{2.35}$$

$$Z = \sum_{i=1...n, j=1...n, i\neq j} z_{ij} I_{ij} + \sum_{i=1..n} z_{ii} J_i \tag{2.36}$$

Since

$$J_i = [I_{ij}, I_{ji}], \quad j \in 1...n, \quad j \neq i \tag{2.37}$$

$$Z = \sum_{i=1..n, j=1...n, i\neq j} z_{ij} I_{ij} + \sum_{i=1...n-1, k\neq i, k\in 1...n} z_{ii} [I_{ik}, I_{ki}] \tag{2.38}$$

So any matrix belonging to $sl(n)$ can be generated from linear combinations of $\mathcal{B}$ and $[\mathcal{B}, \mathcal{B}]$.

This shows that with the non-holonomic constraints, W can still be moved freely in the n^2 dimensional vector space, and thus we will not miss any possible solution.

2.3.4 EM algorithm

The EM(Estimate and Minimize) algorithm is widely used in image deconvolution. The algorithm can achieve good result in image processing because the distribution of all image can fit into one model. This is not true for sound signals can not. H. Attias applied the EM idea to develop the so called independent factor method [6].

The EM method is based on maximizing the log-likelihood with respect to the parameters of the generative model describing those data. In the separation model, instead of the likelihood $E[\log p(y|W)]$, we consider the likelihood of complete data $E[\log p(y, x, q|W)]$, where q denotes the parameters of a distribution model.

The algorithm consist of two steps in one iteration[6]:

(E) Given the observed data and the current model, calculate the expected value of the complete-data likelihood.

(M)Minimize the error function, i.e. maximize the corresponding averaged likelihood with respect to W.

A more generalized EM algorithm making use of ICA is the Seesaw algorithm:

1. Fix the parameter of the generative model, use one or several iterations of ICA rules, then update the posterior estimation using the new separating matrix W.

2. Fix the separation matrix, use single step of EM followed by updating the posterior estimation using the new value of parameters in generative model.

With the separation matrix fixed, the source signal can be reconstructed from the sensor signal. In the existence of noise, a Least Mean Square or Maximum aposterior probability estimator can be used to recover the source signal.

2.3.5 Stability analysis

A stationary point of the algorithm in the above algorithms is characterized by the fact that the update of W has zero-mean. Therefore any invertible stationary point should be a solution of

$$E[G(Wx)] = E[G(y)] = 0 \tag{2.39}$$

The separating stationary points are characterized as follows. Starting with the regular case, let

$$\Lambda_r = diag(\lambda_1, ..., \lambda_n) \tag{2.40}$$

with each scalar λ_i being a solution of

$$E[\varphi_i(\lambda_i s_i)\lambda_i s_i] = 1, \quad i = 1, ..., n \tag{2.41}$$

Because of the independence assumption and the zero-mean assumption, it is then easily checked,that $E[G_r(\Lambda_r s)] = 0$. Therefore the matrix $W = \lambda_r A^{-1}$ is such that $y = Wx = \lambda_r s$ is a separating matrix and is a stationary point of equation 2.31 with $W(t+1) = W(t) = W_o$.

With the stationary points characterized we now study their stability. If these stationary points are not stable they can not be attractors in terms of specific moments of the source distributions. The definition of these moments depends on the non-linear functions used in the function G.

Asymptotic analysis yields two types of stability conditions. The first type is a source-wise condition expressing that the estimation of the scale of each source should be stable. The second type of conditions is pair-wise. The scale stability condition for the ith source is found to be:

$$1 + E[\varphi_i'(y_i)y_i^2] > 0, \quad 1 \leq i \geq n \tag{2.42}$$

One often uses non-linear functions φ_i which are non-decreasing and thus have a positive derivative. In this case, the scale stability conditions are readily met. The pair-wise stability conditions turn out to depend on the non-linear functions φ_i's and on the distributions of the sources via the moments. Define:

$$k_i = E[\varphi_i'(y_i)]E[y_i^2] - E[\varphi_i(y_i)y_i] \tag{2.43}$$

According to Jean-Francois Cardoso [7], The pair-wise stability conditions are:

$$\begin{aligned} (1+k_i)(1+k_j) &> 1, \quad 1 \leq i < j \geq n && (2.44)\\ 1+k_i &> 0, \quad 1 \leq i \geq n && (2.45) \end{aligned}$$

For the natural gradient learning algorithm, we consider the learning equation in its continuous time version as

$$\dot{W}(t) = \mu(t)[I - \varphi(y(t))y^T(t)]W(t) \tag{2.46}$$

where $\dot{W}$ denotes time derivative of the matrix $W(t)$. We consider the expected version of the learning equation

$$\dot{W}(t) = \mu(t)E[I - \varphi(y(t))y^T(t)]W(t) \tag{2.47}$$

By linearizing it at the equilibrium point, we have the variational equation

$$\delta\dot{W}(t) = \mu(t)\frac{\partial E[I - \varphi(y(t))y^T(t)]W}{\partial W}\delta W \tag{2.48}$$

Only when all the eigenvalues of the operator $(\partial E[I - \varphi(y(t))y^T(t)]W)/(\partial W)$ have negative real parts is the equilibrium asymptotically stable. Therefore, we need to evaluate all the eigenvalues of the operator. Since $I - \varphi(y(t))y^T(t)$ is derived from the gradient dl as in 2.31, we need to calculate its Hessian d^2l

$$d^2l = \sum \frac{\partial L(y,W)}{\partial w_{ij}\partial wkl}dw_{ij}dw_{kl} \tag{2.49}$$

in terms of dX. The equilibrium is stable if and only if the expectation of the above quadratic form is positive definite. Define

$$\sigma_i^2 = E[y_i^2] \tag{2.50}$$

$$k_i = E[\dot{\varphi}_i(y_i)] \tag{2.51}$$

$$m_i = E[y_i^2\dot{\varphi}(y_i)] \quad \text{where} \quad \dot{\varphi} = d\varphi/dy \tag{2.52}$$

The separating solution is a stable equilibrium of the learning equation if and only if [2]:

$$m_i + 1 > 0 \tag{2.53}$$

$$k_i > 0 \tag{2.54}$$

$$\sigma_i^2\sigma_j^2 k_i k_j > 1 \tag{2.55}$$

for all i, j, where $i \neq j$.

It is not difficult to show that the stationary point does exist. However, global convergence results are not available for $n >= 2$ case.

2.4 Variation in modeling

All the above methods assume an instantaneous linear mixing model. However, this model does not truly reflect how sound transfers and mixes when it travels through air. In this section, we will explore more complex models that simulate this process more truthfully.

2.4.1 How sound spread in closed space

At normal pressure and standard conditions of humidity, the speed of sound is a function of temperature[13]:

$$s = (331.4 + 0.6t) \quad m/s \tag{2.56}$$

In a typical sized room, successive reflections are too close together to be audible as separate events. For a mid-frequency sound wave with given source and receiver position the room acoustic effect can be seen as a linear time invariant system producing a sum of attenuated, filtered, and delayed versions of the original signal.

Now we consider the intensity of the reflected signal. Intensity is defined as the amount of sound energy flowing across a unit area surface in a second and follows an inverse square law with distance. Thus reflected sound is weaker than sound coming from the source directly because it travels longer distance.

Consider how sound spreads in a room. The audio character of the room is decided by the shape and layout of the room, the position and direction of both the source and receivers, and the absorption characteristics of the boundaries. Most materials absorb more high frequency energy than low frequency energy and thus the reflected sound is a low-pass filtered version of the original signal.

The instantaneous mixing model does not show the room acoustics exactly, that's why we need to look at convolution model.

2.4.2 Filter convolution model

The most common model [5] is one in which the observation $x(k)$ is assumed to be produced from $s(k)$ as

$$x(k) = \sum_{p=-\infty}^{\infty} H_p s(k-p) \tag{2.57}$$

where H_p is an $n \times m$ dimensional matrix of unknown mixing coefficients at lag p. The goal is to calculate $y(k) = [y_1(k) \cdots y_m(k)]^T$ of possibly scaled and delayed estimates of the source signals in $s(k)$ from $x(k)$ using a causal FIR filter given by

$$y(k) = \sum_{p=0}^{L} W_p(k)x(k-p) \tag{2.58}$$

where $W_p(k), \quad 0 \le p \le L$ is a $(m \times n)$ dimensional matrix. We have $W(z)H(z) = P\Lambda(z)$ where P is a permutation matrix and $\Lambda(z)$ is a diagonal matrix with $\lambda_i z^{-\tau_i}$ on the diagonal entry.

2.4.3 State space model

A state space model is intended to express convolution in real world. Let

$$\begin{aligned} X(k+1) &= AX(k) + BS(k) + Pn(k) && (2.59)\\ u(k) &= CX(k) + DS(k) && (2.60)\end{aligned}$$

The transfer function is

$$H(z) = C(zI - A)^{-1}B + D \tag{2.61}$$

and the demixing model is

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + Ln(k) && (2.62)\\ Y(k) &= Cx(k) + Du(k) && (2.63)\end{aligned}$$

The transfer function of the demixing system is

$$W(z) = C(zI - A)^{-1}B + D \tag{2.64}$$

As we can see, it's more general than the convolution model mentioned before in section 2.4.2[20].

2.5 Blind deconvolution

It has been shown that BSS/BSD problems are closely related in structure[11]. The similarity between BSS/BSD makes it possible to solve BSD problem using ideas and methods for BSS.

Consider the n-dimensional source separation task with H as a circulant matrix with the first row

$$H_1 = [h_0 \cdots h_{-M} \quad 0 \cdots 0 \quad h_M \cdots h_1] \tag{2.65}$$

Then

$$x_i(k) = \sum_{p=-M}^{M} h_p s_{[i-p]_n}(k) \tag{2.66}$$

where $[.]_n$ denotes the mod-n operation. Thus $x(k)$ is obtained from the circular convolution of the channel impulse response h_j,$-M \leq j \leq M$, and the source sequence. To extract the source sequence, define a demxing matrix $W(k)$ as

$$W(k)_{ij} = \begin{cases} w_{i-j}(k) & \text{if} \quad [|i-j|]_n \leq L \\ 0 & \text{otherwise} \end{cases} \tag{2.67}$$

The goal is to adjust $W(k)$ such that

$$\lim_{k \to \infty} W(k)H = P\Lambda \tag{2.68}$$

where P is a permutation matrix with a single entry in any row or column and Λ is a diagonal nonsingular matrix.

Since the product of two circulant matrices is also a circulant matrix, $W(k)$ as defined above is adequate to separate the source sequence.

As the dimension of $H(z)$ goes to infinity, the central part of circulant matrix becomes a Toepliz matrix so circulant convolution becomes convolution as long as the sequence defining the matrices are absolutely summable.

Hence

$$x_i(k) = \sum_{j=-\infty}^{\infty} h_j s(k+i-j) \tag{2.69}$$

$$y_i(k) = \sum_{j=-\infty}^{\infty} w_j(k) x_{i-j}(k) \tag{2.70}$$

This is a model for BSS problem. Thus with proper changes to the blind separation algorithm, we can get an algorithm to solve the single channel blind channel deconvolution problem.

If the problem is considered in the frequency domain, convolution becomes multiplication and the deconvolution problem in time domain becomes an instantaneous demixing problem in the frequency domain. That is, for

$$X(f) = A_f S(f) \tag{2.71}$$

We are looking for W_f such that

$$Y(f) = W_f x(f) \tag{2.72}$$

is the closest estimation of $S(f)$. In section 3.3.3 we will derive a deconvolution algorithm from this idea.

2.5.1 Learning under state-space model

The state-space model can also be used to describe a blind separation and deconvolution system. Although theoretically transfer function models are equivalent to state-space models, it is difficult to exploit any common features that may be present in real dynamic systems by using transfer function. State-space models also make it much easier to deal with the stability problem and the realization problem and enable more general descriptions than FIR filtering.

Suppose the mixing model is described by a stable linear state discrete system

$$\begin{aligned} \mathcal{X}(k+1) &= \bar{A}X(k) + \bar{B}S(k) + (k) && (2.73) \\ u(k) &= \bar{C}\bar{x}(k) + \bar{D}s(k) + \theta(k) && (2.74) \end{aligned}$$

where s(k) is a m-dimensional source vector, u(k) is the n-dimensional sensor signals, $\bar{x}$ is the state vector of the system, $\xi_P(k)$ is the process noise, and $\theta(k)$ is the sensor noise of the mixing system. The transfer function of the system without noise is

$$H(z) = \bar{C}(zI - \bar{A})^{-1}\bar{B} + \bar{D} \tag{2.75}$$

The demixing model is another linear state-space system

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + L\xi_R(k) && (2.76) \\ y(k) &= Cx(k) + Du(k) && (2.77) \end{aligned}$$

where $\xi_R(k)$ is the reference model noise. The goal is to adjust A,B,C,D,L such that

$$W(z) = C(zI - A)^{-1}B + Dy(k) = W(z)H(z) = P\Lambda(z) \tag{2.78}$$

where P is a permutation matrix and $\Lambda(z)$ is a diagonal matrix with $\lambda_i z^{-\tau_i}$ as the diagonal elements.

If the matrix $\bar{D}$ is full rank, the inverse system exists.

Parameter C, D can be learned with same scheme as described in the FIR filter model. We estimate the state vector based on Kalman filter method instead of directly adjusting the A, B matrices.

$$x(k+1) = Ax(k) + Bu(k) + Kr(k) + \xi_R(k) \tag{2.79}$$

where K is the Kalman filter gain matrix, r(k) is the innovation vector. In this case no explicit residual r(k) is available because the expected output $y(k)$ should be the original signal which is unknown. Here we use an estimation technique known as hidden innovation defined by

$$r(k) = \Delta y(k) = \Delta C x(k) + \Delta D u(k) \tag{2.80}$$

The hidden innovation indicates the direction to adjust the output of the demixing system and is used to generate an aposterior state estimate. Now the common Kalman filter can be used as follows to estimate the state vector $x(k)$.

1. Compute the Kalman gain matrix

$$K(k) = P(k)C(k)^T[C(k)P(k)C^T(k) + R(k)]^{-1} \tag{2.81}$$

2. Update the state vector using the hidden innovation

$$\widehat{x}(k) = x(k) + K(k)r(k) \tag{2.82}$$

3. Update the error covariance matrix

$$\widehat{P}(k) = [I - K(k)C(k)]P(k) \tag{2.83}$$

4. Evaluate the error covariance matrix ahead

$$P(k) = A(k)\widehat{P}(k)A(k)^T + Q(k) \tag{2.84}$$

   with initial condition P(0)=I, where I is the identity matrix. Here $Q(k)$, $R(k)$ are the covariance matrices of the noise vector ξ_R and output measurement noise n_k respectively.

The state-space idea is appealing to people in the field of controls because it makes clever use of the Kalman filter. However, the idea is expensive in computation and thus not practical in real situations.

2.5.2 Learning under the filter deconvolution model

Generally speaking the same cost function can be used for both the separation problem and the deconvolution problem as long as the filter coefficient sequence has a bounded L_2 norm.

The following are different algorithms derived from different error functions:

1. Minimize mutual information between outputs

$$I(Y, X) = H(Y) - H(Y/X) \tag{2.85}$$

   where $I(Y, X)$ is the mutual information contained in the output Y about the input X, $H(y)$ is the entropy of Y.

   $H(Y/X)$ is the entropy of the output Y that did not come from the input X. Then

$$\frac{\partial}{\partial \omega} I(Y, X) = \frac{\partial}{\partial \omega} H(Y) \tag{2.86}$$

   since $H(Y/X)$ does not depend on ω.

2. Minimize the Kullback divergence between the output and the product of marginal output

$$D_{f_Y \| \widetilde{f}_Y} = \int_{-\infty}^{\infty} f_Y(Y) \log(\frac{f_Y(Y)}{\prod_{i=1}^{n} \widetilde{f}_{y_i}(y_i)}) = -H(Y) + \sum_{i=1}^{n} \widetilde{h}(y_i)) \tag{2.87}$$

   where $H(\cdot)$ is the entropy of a probability function.

3. Maximum likelihood [6]

$$\theta = [A, n], \quad \theta^{-1} = [A^{-1}, -A^{-1}n] \tag{2.88}$$

   where n is observation noise.

$$p_x(x, \theta) = \mid detA \mid^{-1} g[A^{-1}(x - n)] \tag{2.89}$$

According to S. Amari, S. C. Douglas, A. Cichocki and H. H. Yang [5], the most effective form is

$$J(w(z,k)) = -\sum_{i=1}^{m} \log p_i(y_i(k)) - \frac{1}{2\pi j}\oint \log |detW(z,k)| z^{-1} dz$$

Define

$$\varphi_i(y_i) = -\frac{d}{dy_i}\log p_i(y_i)$$

Then

$$d(-\sum_{i=1}^{m}\log p_i(y_i(k))) = \sum_{i=1}^{m}\varphi_i(y_i(k))dy_i(k) = \varphi^T(y(k))dx(z,k)y(k)$$

Similarly

$$d(\frac{1}{2\pi j}\oint \log |detW(z,k)|z^{-1}dz) = \frac{1}{2\pi j}\oint tr(dW(z,k)W^{-1}(z,k))z^{-1}dz = tr(dX_0(k))$$

Thus the natrual gradient deconvolution algorithm is

$$W_p(k+1) = W_p(k) + \mu[W_p(k) - \varphi(y(k))u_p^T(k)] \tag{2.90}$$

with

$$u_p(k) = \sum_{q=-\infty}^{\infty} W_q^T(k)y(k-p+q) \tag{2.91}$$

In a practical implementation, one can approximate the doubly infinite filter with a FIR causal filter given by

$$y(k) = \sum_{p=0}^{L} W_p(k)x(k-p) \tag{2.92}$$

We now have

$$W_p(k+1) = W_p(k) + \mu[W_p(k) - \varphi(y(k-L))u^{*T}(k-p)] \tag{2.93}$$

with

$$u(k) = \sum_{q=0}^{L} W_{L-q}^{*T}(k)y(k+q) \tag{2.94}$$

We now show the same equation 2.93 can be deducted from the natural gradient ICA in the frequency domain. First we have

$$X(f) = A_f S(f) \tag{2.95}$$

Where $X(f)$ and $S(f)$ are the Fourier transform of the mixtures and the original sources at frequency f. We are looking for W_f such that

$$Y(f) = W_f X(f) \tag{2.96}$$

is the closest estimation of $S(f)$.

The learning rule from the natural gradient algorithm [4] is

$$\Delta W = \mu[I - \varphi(y)y^{*T}]W \tag{2.97}$$

Rewriting the same equation in the frequency domain we have

$$\Delta W_f = \mu[I - fft(\varphi(y))Y_f^{*T}]W_f \tag{2.98}$$

Since the assumption of independence is also valid in the frequency domain, all the deductions we have done before are still valid and the frequency domain ICA algorithm has the same form as the above equation 2.93.

Now apply the Fourier transform to both sides of 2.98 to get

$$\Delta W(z) = \mu[W(z) - \varphi(y) * y^{*T} * W(z)] \tag{2.99}$$

With

$$\begin{aligned} u_p &= [y(t-p)^{*T} * W(z)]^{*T} && (2.100) \\ &= \sum W_q^T y(k-p+q) && (2.101) \end{aligned}$$

We have

$$\Delta W(z) = \mu[W(z) - \sum \varphi(y-L) * u_p^{*T}] \tag{2.102}$$

This is exactly the same format in [5] as mentioned above in 2.93.

2.5.3 Kuicnet approach

According to the Central Limit Theorem, the sum of n independent identically distributed random variables has a Gaussian distribution as n goes to infinity. For n independent but not identically distributed random variables, if their PMF's satisfy certain conditions, then the central limit theorem still holds. Define kurtosis of a random variable as

$$E(y(k)^4) - 3(E(y(k)^2)^2) \tag{2.103}$$

Obviously a Gaussian random variable has kurtosis equal to 0. One can verify that the closer the kurtosis is to 0, the closer the random variable is to a Gaussian random variable. Kurtosis can then be used as a cost function for separation and deconvolution.

For the single channel problem[12] we have

$$w(k+1) = w(k) + \mu[|y(k-L)|^2 y(k-L)u^*(k) - |y(k-L)|^4 w(k)] \tag{2.104}$$

where

$$u(k) = \sum_{j=0}^{L} w_{L-j}^*(k) y(k-j) \tag{2.105}$$

Alternatively

$$w(k+1) = w(k) + \mu[y^*(k) f^T(y(k)) - F(y(k))]w(k) \tag{2.106}$$

where

$$\begin{aligned} y(k) &= [y(k) \cdots y(k-L)]^T && (2.107) \\ F(y(k)) &= \{diag|y(k)|^4, \cdots, |y(k-L)|^4\} && (2.108) \\ f(y(k)) &= [|y(k)|^2 y(k) \cdots |y(k-L)|^2 y(k-L)]^T && (2.109) \end{aligned}$$

Chapter 3

Sub-band Based ICA Algorithm

In last chapter we introduced several methods of blind separation and blind deconvolution. By theoretical deduction and experimental result we showed that the most effective method is the non-holonomic algorithm proposed by Shunichi Amari, T. P. Chen and A. Chichocki.

In this chapter the algorithm is further developed by making use of wavelets. A more robust and faster algorithm, namely the sub-band ICA algorithm is presented. In section 1, a brief introduction to the human hearing system is given. This system is the inspiration for the sub-band ICA algorithm. In s section 2, the process of sub-band ICA is described in detail. In section 3, several problems of the sub-band ICA implementation are discussed. The sub-band ICA algorithm is extended to solve the deconvolution problem.

The idea of subband-based ICA idea comes from the fact that humans process acoustic signals on different frequency bands independently. This method developed by Yuan Qi, P.S. Krishnaprasad and S. Shamma [18] provides robust performance in the presence of noise and reduces the computational complexity. This idea enables a real-time separation method.

3.1 Filter-bank structure in the human ear

It is well known that sound waveform spreading in the air is transformed in vibrational mechanical energy in the middle ear, then further in to the vibration of the basilar membrane located inside the cochlea. Since the basilar membrane is narrow and stiff near the base, and wider and softer near the end. High frequencies excite the base portion more strongly than the end portion; on the contrary low frequencies excite the base portion more weakly than the end portion. High and low frequency disturbances arrive at their respective peak basilar membrane points nearly simultaneously. This leads to the assumption that the basilar membrane acts like a filter bank. The vibration of the basilar membrane produces motion of the stereocilia which then cause the response of the auditory nerves. Thus the entire process of human hearing starts with a filtering action. The engineering model of ear is showed in Figure 3.1

Much work has been done to develop the bank model. For example, in [13], Dudley used a bank of ten bandpass filters, each with a frequency width s of 300Hz, to process a human voice ranging from 300-3000Hz.

3.2 Sub-band ICA

The outline of the algorithm is described as the following:

1. Each component, $x_j(n)$, of the observation $x(n)$ is filtered through a filter bank, resulting in a subband signal. Two possible choices of the filter bank are a cochlea filter or an orthogonal Daubechies wavelet packet. Since wavelet packet is easier to implement and provide linearity, the Daubechies wavelet packet is used in the final implementation.

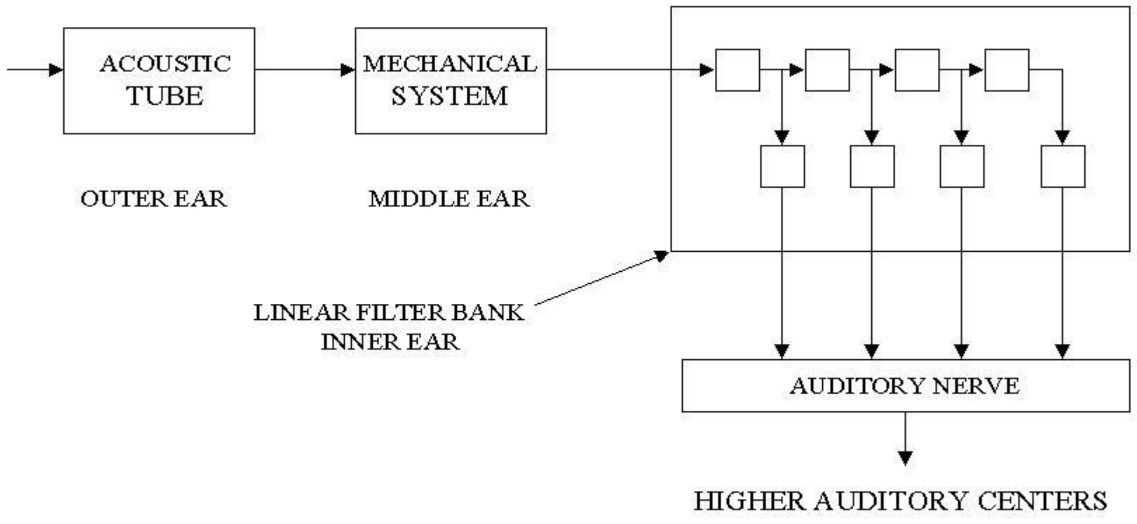

Figure 3.1: The model of human ear

2. The power of the decomposed signal in each band is computed and ordered.

3. The ICA learning algorithm is applied on the bands with the strongest power

4. The soft-thresholding algorithm is applied to the subband decomposed signals.

5. The overall demixing matrix W is received from the demixing matrix of each of the subbands, by using the competitive learning rule to cluster the rows of the demixing matrices on different sub-bands.

Figure A.3 shows the structure of the sub-band ICA method.

The sub-band ICA algorithm improves the performance of ICA in the presence of noise. If the noise is narrow-band, then good separation can be performed on the noise free sub-bands. If the noise is broad band, ICA is performed on those sub-bands with strongest power, i.e. largest signal to noise ratio(SNR).
Since the wavelet coefficients are typically Laplace distributed, the sub-band signal has a more peaky and heaviky tailed distribution than the original signal. And thus the sub-band based ICA converges to the demixing matrix with faster speed. The sub-band based separation idea has also been proposed by Hyung-Min Park [17]. In Park's work the learning is performed in the frequency domain and actually incooparates the convolution model and the idea of speech recognition. The method proposed by Yuan Qi, P.S. Krishnaprasad and S. Shamma is more general, and less computationally demanding.

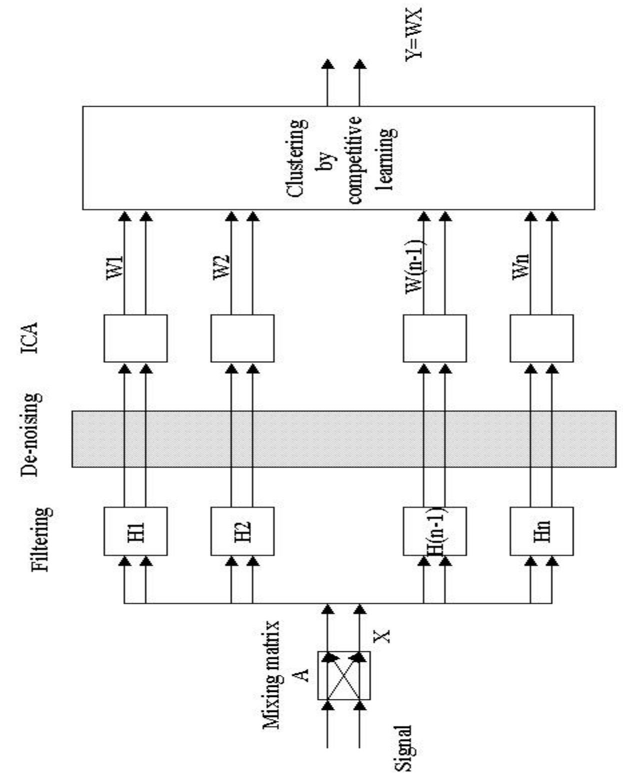

Figure 3.2: The block diagram of Subband ICA algorithm

3.3 Implementation of the algorithm

3.3.1 Use adaptive basis selection in the wavelet packet

Sub-band ICA enhances the separation capability of standard ICA by decompositing the signal into different frequency bands. The design of the filter bank can greatly affect the performance. For example, we should not divide the signal into two bands where it should be continuous in the time-frequency plane. The filter bank design should vary according to different signal properties. The solution to this problem is to apply the adaptive basis selection algorithm [8] on the summation of all the mixed signals to get the best bases. This algorithm ensures a filter bank that does not not split any of the signals into improper frequency bands.

3.3.2 Selecting the bands to perform ICA

The ICA algorithm does not need to be applied on every band coming out of the filter bank. In reality, the mixing matrix for different frequency bands varies from one to the other. In BSS, however, only one mixing matrix is used in the model. Estimating the mixing matrix of the bands with strongest power and then taking the average will produce the best separation result. From the experiment on human voice, we find the strongest one-quarter of the total bands always contain around sixty percent of total energy. For music signals the energy spread more evenly, but one-quarter to one-half of total bands is still able to produce good separation result. The amount of computation is greatly reduced.

3.3.3 Introducing the convolution model into the sub-band algorithm

The convolution model provides a much better simulation of how sounds are mixed in the process of traveling. By introducing this model into the sub-band algorithm we can expect it to produce better separation performance, although with a heavier computation load.

THus after sub-band filtering, instead of applying the non-holonomic ICA algorithm, we apply the natural gradient algorithm describe in Chapter 2,

$$W_p(k+1) = W_p(k) + \mu(k)[W_p(k) - f(y(k-L)u^{*T}(k-p)] \tag{3.1}$$

where

$$u(k) = \sum_{q=0}^{L} W_{L-Q}^{*T}(k)y(k-q) \tag{3.2}$$

The mixing filter W for signals in different frequency bands are not exactly the same. According to Ben Gold[13], the surface of any object has different reflection absorption characteristics to sound waves of different frequencies. In addition, waves of different frequencies show different diffusion when there is an obstacle in the path of transmission. So instead of taking a weighted average on the demixing matrices to get an overall demxing matrix as we did in sub-band ICA, we should separate the mixture in each sub-band by it's own demixing matrix and then recover the original signal by going through an inverse filter bank. Unfortunately, this method is too complex in computation. The experiment shows that applying the deconvolution filter for the sub-band with highest energy will work reasonabley well to recover the original signals.

Chapter 4

Working with Real World Signal

In Chapter 2 and Chapter 3, the theoretical aspects of BSS/BSD algorithms were discussed. Here we explore the implementation side. An integrated system, including a mobile sensor platform and computation unit is built, and a real time ICA algorithm is implemented on the system to process signals recorded by the sensors. In Section 1 the hardware setting and environment are described; in Section 2 specific problems related to real-time processing are discussed; in Section 3 a new criterion to evaluate the performance of separation without knowing the mixing matrix is introduced and verified.

4.1 Hardware and environment

To give an overview of the hardware environment in this project, we first mention the hardware used in the sequence of data flow:

1. Styrofoam head equipped with two microphones and amplifiers. The microphones are mounted approximately at the ears and are used to collect sound data.

2. Nomadic Technology Super Scout II mobile robot. This is the mobile platform that carries the sensors and transfers data to computing unit.

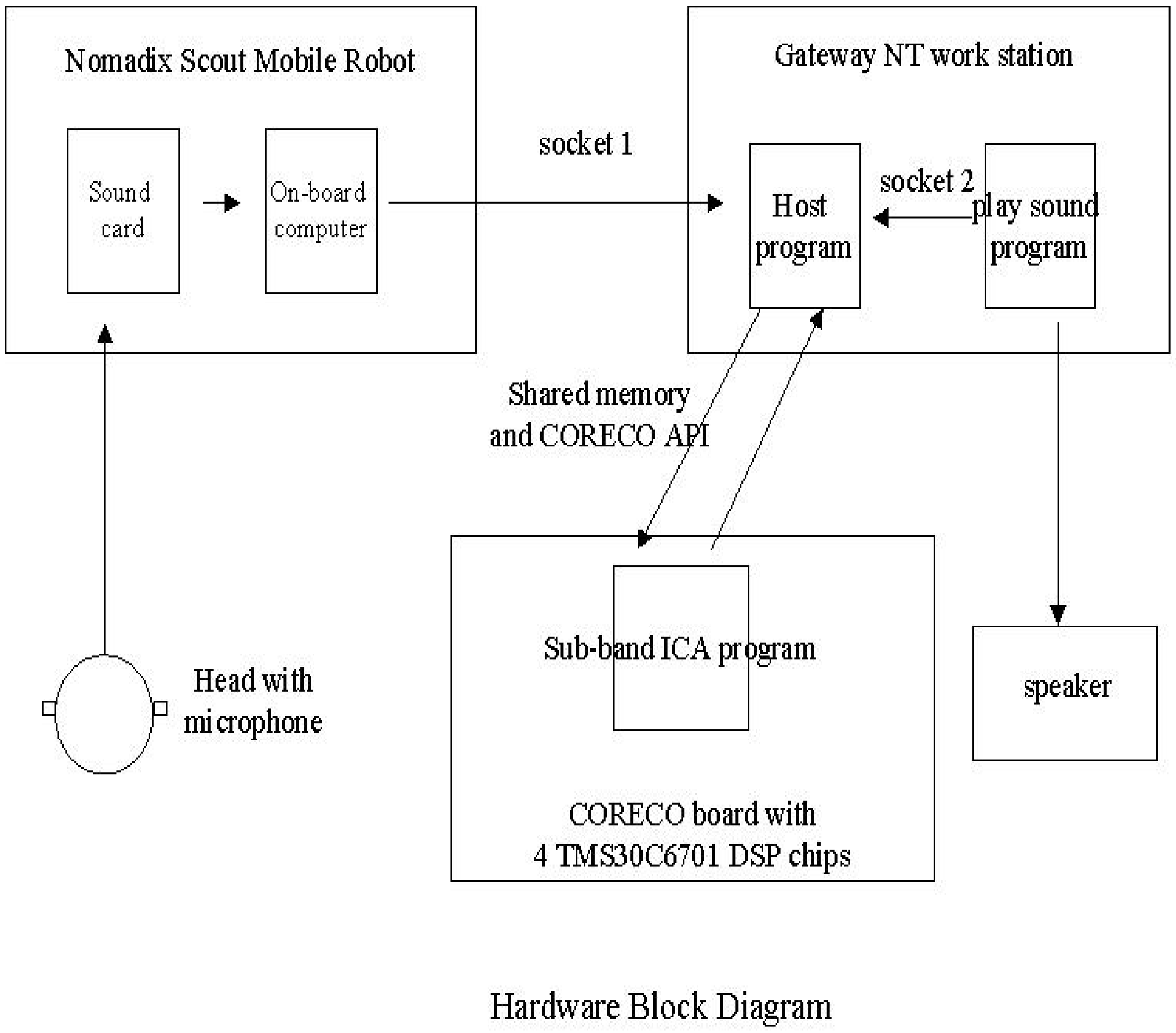

Figure 4.1: The block diagram of the hardware setting for this project

The capability and functions of the hardware components are described respectively below.

4.1.1 Styrofoam head and robot

The task of collecting data is performed by Nomadic Super Scout II mobile robot. A Styrofoam head is mounted on the robot. In the position of its ears, two microphones are installed to simulate the human auditory perception. A specially designed amplifier is used to transfer the signal from the microphones to a proper level for the sound card on the robot. The amplifier needs a DC power supply that can either be provided by batteries or the power supply of robot.

Data is sampled at a rate of 8KHz, and is transfered as blocks containing 512 data elements. Each data element is represented in PCM format as a 16 bit signed integer.

The robot is equipped with a set of touch sensors and sonar sensors, and a dedicated board to control the motors and sensors. At the top level is a PC/104 embedded PC. RedHat Linux runs as the host operating system. The robot is connected to a wireless network via an IEEE 802.11 network card.

4.1.2 Windows NT workstation

Windows NT provide a versatile platform to cooperate among the console application, TI code composer, the robot and MATLAB. In this project, Windows NT acts as the bridge between data collector and the computing unit. The workstation here is Gateway E-5200 machine with the Windows NT operating system and will be called “the host” in the following. A console program runs on the Windows NT machine, which communicate between the robot and DSP processor. It is referred to as the “host program”.

4.1.3 DSP processor

The DSP we use in this project is the TMX320C6701 float point digital signal processor, which has a 167 Mhz clock rate and allows four float point arithmetics, two fixed point arithmetics and two multiplications to happen at the same time. The processing power helps the complicated ICA algorithm to be implemented in real time.

In the design of this project, the DSP processors are responsible for executing the ICA or deconvolution algorithm, denoising the input data, and calculating the separated output. The computation task is divided between two DSP processors.

The master DSP perform the following tasks: communicate with the host application; reads the input data stream; passes the data through a wavelet filter bank; denoises it; puts the intermediate data into shared RAM for the slave DSP to read, and reads the output of the slave DSP when it is ready, ; implements the wavelet reconstruction, multiplies by the separation matrix to produce the output. The slave DSP calculate the separation matrix from the data it finds in the shared RAM and puts the the new separation matrix into the shared RAM for the master DSP to read.

The task of the master DSP is performance critical to ensure a continuous sound output. All the computation has to be performed at a speed faster than the the speed at which sound data is collected. If the data processing is not fast enough, some data will be lost. The task of the salve DSP, however, is not that urgent. In a realistic environment, the mixing matrix is not changing very fast. A separation matrix calculated from data collected 1/10 second ago will still work well to separate the current data.

4.1.4 Python board

The Python/C6 multi-DSP board installed on the NT machine can support up to four 6701 DSP chips. A maximum of 4MB of shared-RAM between the C60 can serve for inter-DSP communication.
A set of APIs are provided to communicate between the host machine and the DSP chips. The C60 Native API provides DSP programmers direct control over the resources. The C60 Host API allows an application on the host to handle basic I/O operations with the Python/C6, and must be used in conjunction with a C60 Native API application running on the Python/C6's DSPs.

4.1.5 Room environment

In this experiment, the reflection from the wall and objects in the room cannot be neglected. The robot will be put in the center area of a 10×12 empty space and slowly moved in circle of radius two to three feet. Around the empty space are walls and tables that will scatter and reflect sound. The separated data is played out through a speaker in another area of the room and thus will not produce any interference with the room acoustics.

4.2 Considerations for a real time implementation

All the algorithms we discussed in previous section are designed for off-line experiment. But here we will conduct the experiment in real-time. A continuous output of the separated sound signal is desired, with as little delay as possible. Thus several adjustments to the algorithm need to be made.

4.2.1 Data communication

A program on the robot builds the socket connection with the host program, reads the data from a buffer, and periodically sends the data to the host through the socket.

The host program is responsible for the following tasks: 1. building socket connections with the data collecting process on the robot and the data play-out process; 2. loading the sub-band ICA program on to the DSP chips, and communicating control message and display information to the DSP chips. The main structure of the host program is a loop waiting for command input from console. The processing functions are used as call-back functions from a dynamic link library. When message arrives, such structure produce the fastest response to a message coming from DSP.

4.2.2 Optimizing the code for speed and memory size

Many tools were provided with the DSP to speed up the program. Here we chose to use the following:

1. Write the most computationally demanding part in assembly language. As the wavelet filtering and convolution part is executed on every incoming data block, the task is heavy and time critical and should thus be written in assembly language. Other parts of program are still in C to make the whole project easy to read and manage.

2. Select optimizing parameter. In the compiler provided by TI, there are several choices about how to optimize the code. The program is both time-critical and memory critical and the compiling parameter should be set ac-

cordingly.

3. Make use of the parallel processor. On the the Python there are 4 DSP chips, and each chip has the ability to execute several manipulation concurrently. Evenly dividing the tasks between several DSPs and performing matrix computations in parallel can speed up the process a lot.

The high speed memory for each DSP is small compared to the large amount of data. The strategy is to dynamically allocate and release fast memory inside every cycle. The shared memory among the DSPs is slow so the data exchange between DSPs needs to be carefully designed.

4.2.3 Tuning parameters for the best performance

Much research has been done and many different approaches have been proposed. We have described the reason to choose sub-band ICA with non-holonomic natural gradient algorithm as the core algorithm. Still there are several variables undecided. The choice is made by balancing computational load and performance, and considering the characteristic of the original signals.

1. Choice of non-linear function $\varphi(y)$. The optimal choice is

   $$\varphi_i(y_i) = -dlog(p_i(y_i)))/dy_i$$

   which yields the fastest convergence behavior. Convergence still happens using other functions. For the case of voice a sub-Gaussian signal,

   $$f_i(y_i) = |y_i|^2 y_i$$

   yields adequate separation. For super-Gaussian sources,

   $$f_i(y_i) = tanh(\gamma y_i) \quad with \quad \gamma > 2$$

produce good result.

2. Choice of μ. The learning process should be fast enough to follow the changes in mixing matrix resulted by the movement of robot and thus this parameter cannot be too small. If μ is too large an overflow can result and thus a monitor should be set in program to reset the learning process without causing error.

3. Choice of block size and filter channels. This needs to be considered with the constraint of memory size. In our implementation we chose data block with 512 data elements from each sensor, filtered through 16 channels. The ICA algorithm is applied to four or eight of them with the biggest power intensity.

4.2.4 Smoothing between blocks

The calculation is based on 512 data elements blocks, which corresponds to a sample of 0.064 second. To produce a consistent output sound with good quality, the separation matrix used for each block must not change too much in one step, and must be consistent in scale factor. Two measure were used here to prevent inconsistency between the learning result of two consecutive blocks. First, the separation matrix of the last block was used as the start value of the learning process for the new data block. Second, clustering was used to match the output channels.

4.3 Performance evaluation criterion

In the case where the mixing matrix is known, the performance can be measured by the product of mixing matrix and demixing matrix. Defining $P = WA$, the most commonly used performance index is:

$$E = \sum_{i=1}^{n}(\sum_{j=1}^{n} \frac{|p_{ij}|}{max_k|p_{ik}|} - 1) + \sum_{j=1}^{n}(\sum_{i=1}^{n} \frac{|p_{ij}|}{max_k|p_{kj}|} - 1) \tag{4.1}$$

It is easily seen that perfectly separated signals will have E equal to zero. The smaller the value of E, the better the separation is.

When no information about mixing matrix is available the only clue we have is the output signal. The mixed signals are similar to each other while well separated signals should be different in shape. However, the effect of scaling and delaying must not be considered, because the separation algorithm is not controlled in these two directions. Thus the signals cannot be compared directly in the time domain. One idea is to calculate the statistical distribution of the output signals by counting histogram and comparing the result. This method completely removes scaling and delay factors. However, although different sound sources produce signals with different probability distributions, the distribution functions can be very similar to each other. For example, two speech sentences from two different male human speakers have almost the same distribution curve.

Another idea is to consider the frequency domain. Transforming the signal to the spectral domain can remove most of the delay effect. To remove the scaling effect and reduce computation, a natural idea is that of Linear Prediction Coefficients(LPC). LPC are chosen to minimize the squared error between the observed and predicted signals. The predicted signals tend to be consistent in spectrum with the original signal at the peak but not at the valleys, giving an envelop of the spectrum of the original signal. Using higher order coefficient gives more accurate estimation but is computationally more expensive. For the purpose of speech recognition, a tenth order predictor has the lowest Akaike Information Index(AII). In this project, we need to judge the separation result of signals with a wider spec-

Judged by listening	very successful	good	poor
index E	0.2247	1.3773	3.5344
LPC index	68.6908	0.8948	0.2402

Table 4.1: Comparison of two kind of performance index

trum than speech signal (for example, music), we choose the LPC order to be 20. The normalized difference between LPC of the original signal and the separated signal is a good criterion for the two signal case. Here is a comparison between a commonly used index and a LPC index calculated on separated signals with known mixing matrix in Table 4.1 . We can see the LPC index well represents the quality of separation. The bad thing about it is the index may be very large when we have a very good separation (without noise). That is, it is not normalized, not like the index E as defined in equation (4.1).

Chapter 5

Experimental Results

In this chapter different experiments are described in detail and the results are displayed and compared with separation result that appear in earlier literature. From the analysis of these experimental results, some important conclusion about the ICA model are derived.

5.1 The effect of source type on performance

In the case of two sound sources, we compare the separation results of two male voices, two female voices, one voice and one instrument, and two instruments. The performance of the separation algorithm is affected by how similiar the two sound sources are. The similarities are considered in time domain, frequency domain, intensity level and time delay. Intensity and time delay are determined by the source and microphone position and will be discussed in a later section.

Since the separation algorithm is built from the difference based on the statistical distribution, if the original signals are similar in distribution, the ICA algorithm does not produce good separation results. Off-line MATLAB experiments already

source 1	man's voice	man's voice	man's voice
source 2	man's voice	women's voice	music
index E	0.7133	0.3364	0.2247

Table 5.1: Simulation result of sub-band ICA algorithm on different type of sources

show that separation between music and voices is better than two different voice of the same sex (see Table 5.1).

Our work with real-world recordings has produced results consistent with the

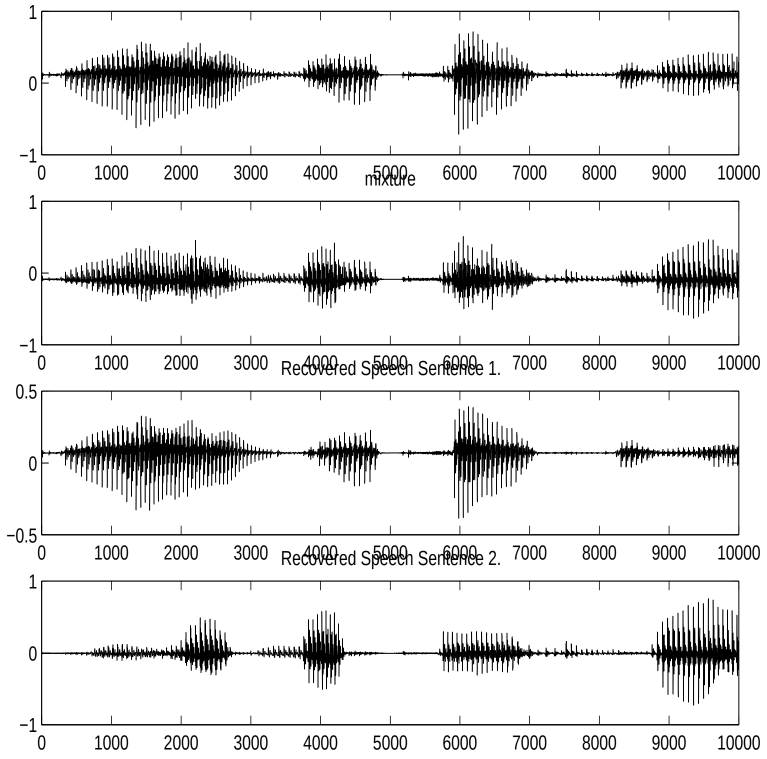

Figure 5.1: The separation of two man's voice

simulations as shown in Table 5.2.

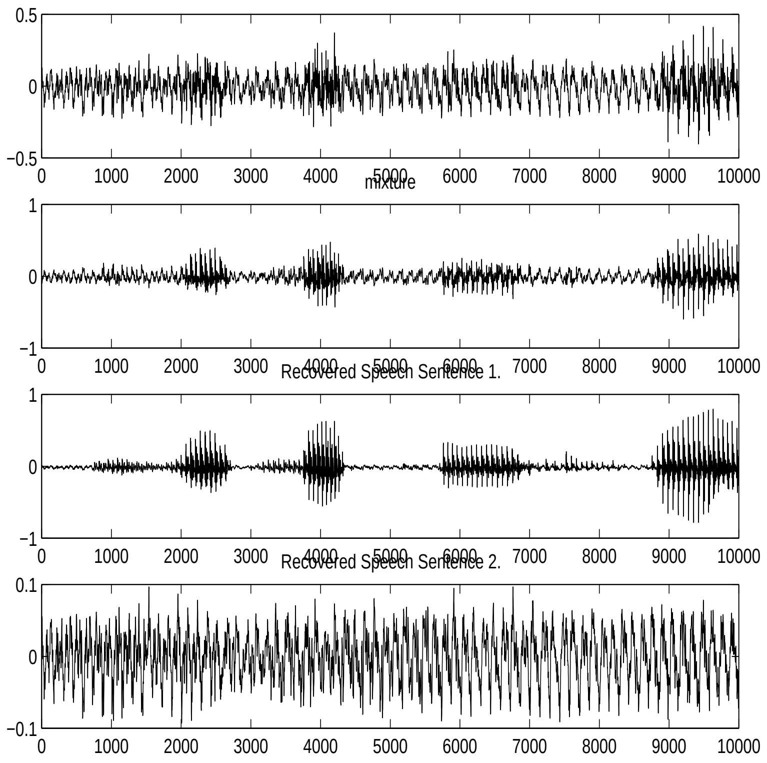

Figure 5.2: The separation of voice and music

source 1	man's voice	man's voice	man's voice
source 2	man's voice	women's voice	music
index E	0.6884	0.3028	0.2133

Table 5.2: Simulation result of real-time separation of real-world recording on different type of sources

5.2 The effect of source distance and angle

In this experiment we want to show that the delay in transmission and reflection can not be omitted for real world signal. That means the instantaneous mixing model can only produce good result under certain condition. The best results can be expected when the position of source and alignment of microphone work together so that both source arrive the two microphone at the same time. This area is shown in grey in Figure 5.3. Unfortunately since the signal from this area arrive

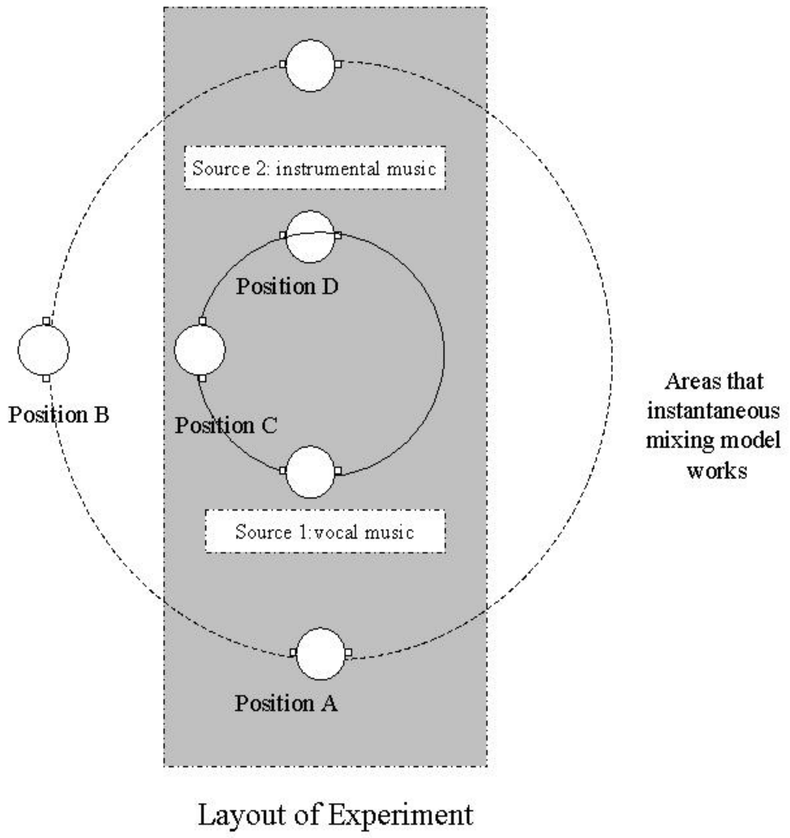

Figure 5.3: The layout of Experiment

at both microphones via similar paths and distances, the intensities do not have much difference either. Thus the signals are mixed by a ill-conditioned matrix. In such situations, separation can be performed but the performance and convergence speed are poor. By comparing the separation output of an instantaneously mixed signal and the real-world mixed signal in position B (same signal source but in different time zone) on Figure 5.4, the disadvantage of instantaneously mixing model can be seen.

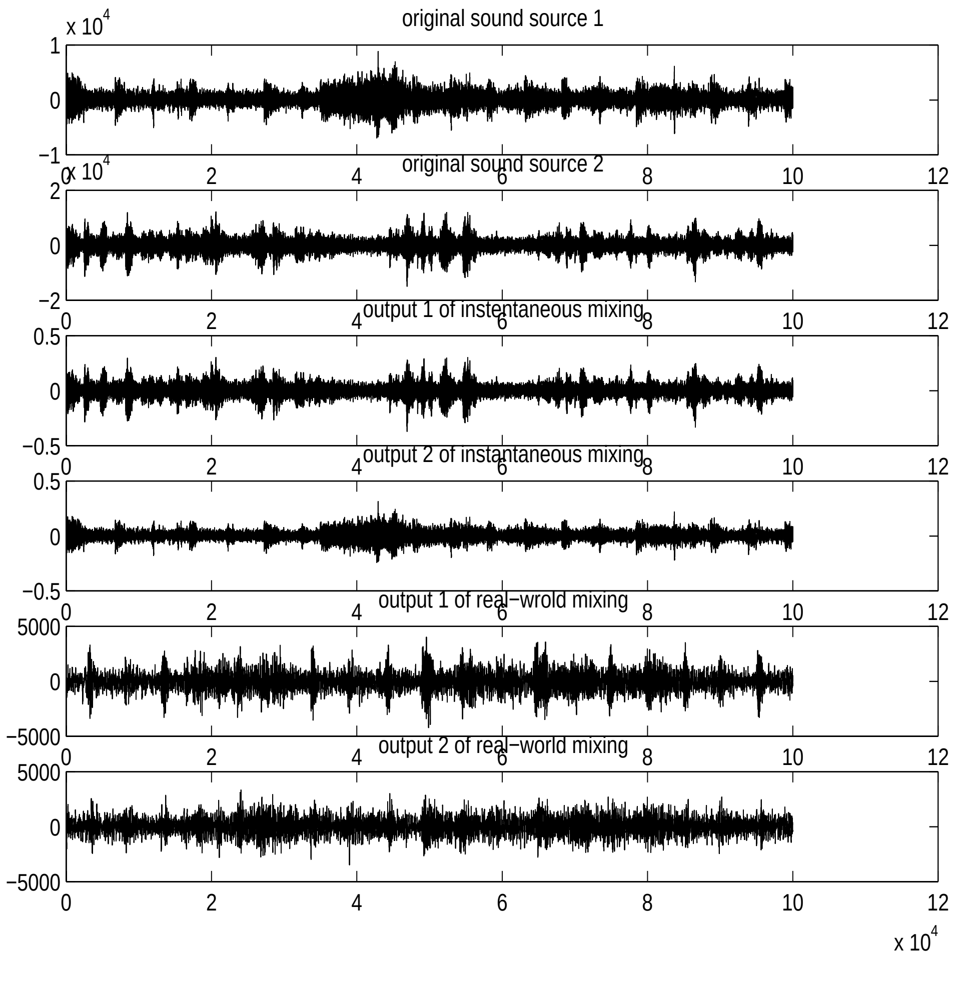

Figure 5.4: Comparing the instantaneous mixing model and the real-world mixing model

When the robot is moving around and sources stay in a fixed position, the quality of the separated signal changes dramatically depending to the robot position. The placement of source and robot are shown in Figure 5.3, and the performance index at four positions are shown in Table 5.3. The output wave form in position A, C are showed in Figure 5.5 and 5.6 respectively.

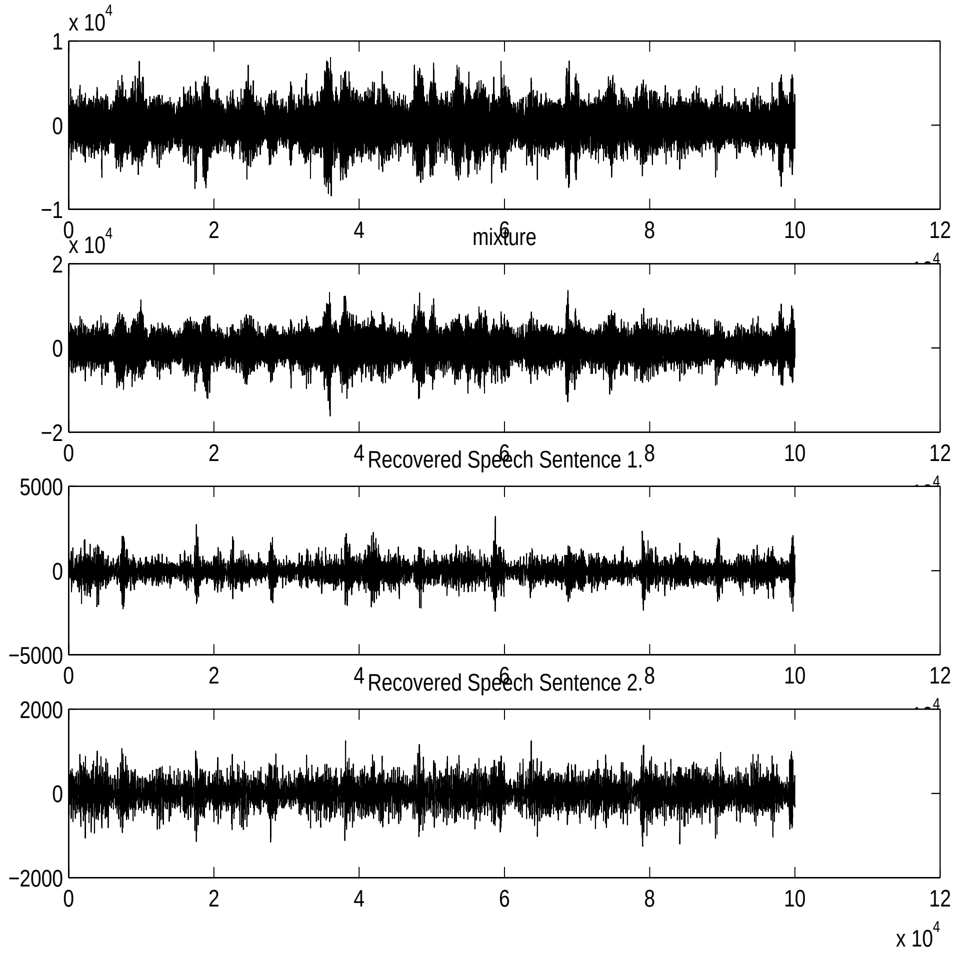

Figure 5.5: Separation of real-world mixture in in position A

In all positions, the separation algorithm is still working, that is, the separated waveforms show the characteristic of a voice signal and a music signal. However, when played out, the acoustic effect is not ideal. The convolution effect is obvious,

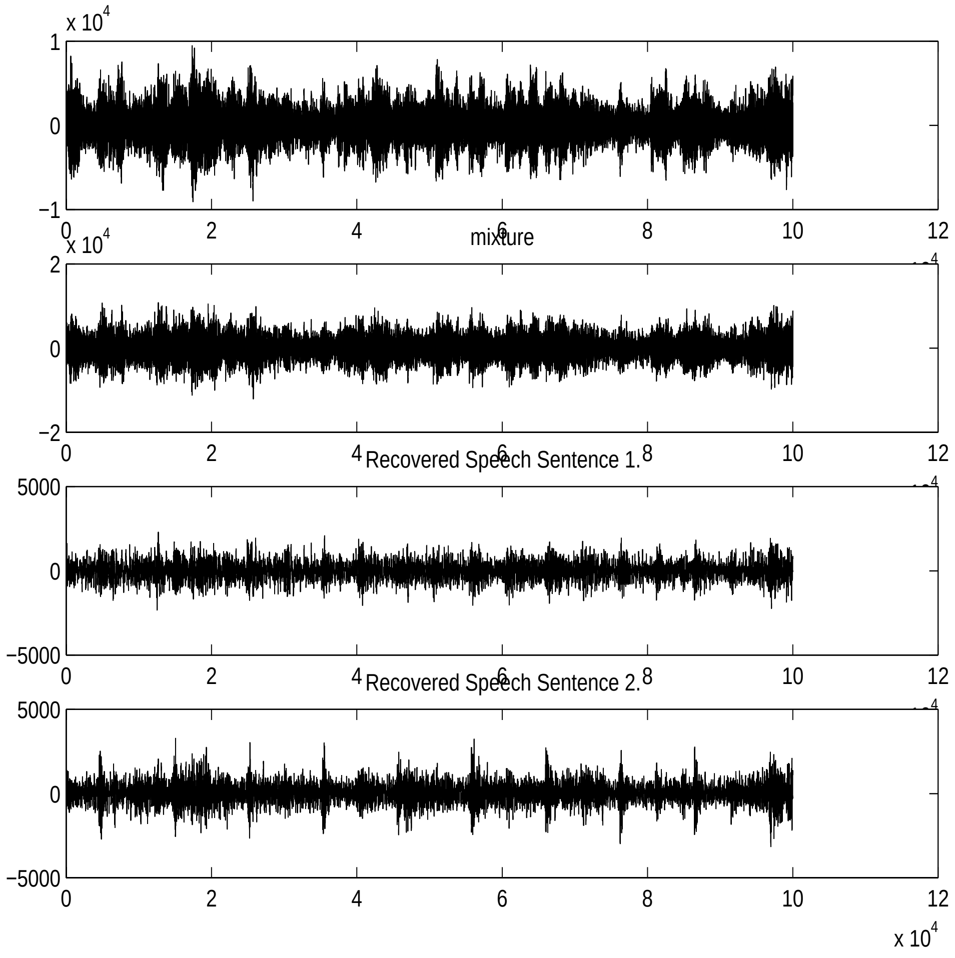

Figure 5.6: Separation of real-world mixture in position B

position	A	B	C	D
LPC index	0.88948	0.0407	0.1953	1.0311

Table 5.3: Performance index of sub-band ICA on real-world recording on different source angle

and the music sounds in a lower tone than it should. Besides the voice signal is not clear.

Thus we need the convolution model to achieve better and more robust performance. With the deconvolution algorithm, separation results show dramatic improvement. The following figure 5.7 shows the output from deconvolution algorithm under the same conditions as described before. The performance index of

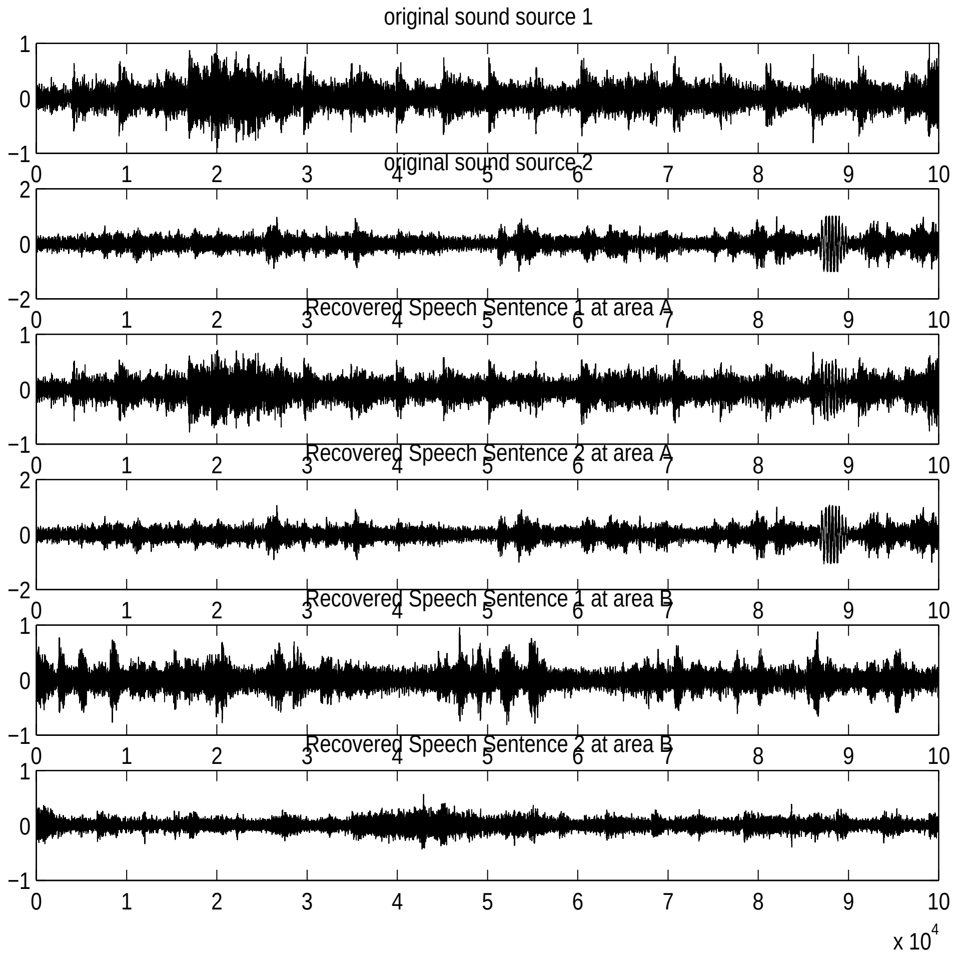

Figure 5.7: Result of deconvolution algorithm in position A and B

sub-band deconvolution algorithm at the same placement of source and robot as Table 5.3 is shown in Table 5.4.

position	A	B	C	D
LPC index	5.4293	6.0226	4.9812	5.3354

Table 5.4: Performance index of sub-band deconvolution on real-world recording on different source angle

5.3 The effect of room acoustics

Even if the experiment is restricted to a closed room and a fixed environment, the mixing filter still change dramatically with repect to the position of the microphones and sources. Reflection has a large effect on the mixing filter. The following plots show the impulse responses of the demixing filter at position B in Figure 5.3. The filter shape is pretty random. It is reasonable since the distance between the two microphones is approximately 10 cm, corresponding to 140 data samples at 8KHz and a 10 tap filter is just too short to show the convolution process. When both sensor and source are placed close to the wall the effect of reflection should be very strong. However, because of the limited computation power and memory, the filter tap in simulation cannot more than 50 while the delay caused by refection is generally much more than that and thus the reflection effect does not appear in the simulations so far.

5.4 Over-complete and under-complete mixtures

In a realistic environment the number of sound sources may change from time to time. When a voice signal is studied, for example, the pause between words and sentences may be long enough that the algorithm should consider the signal source as having been turned off. When there are more sources than microphones, we call

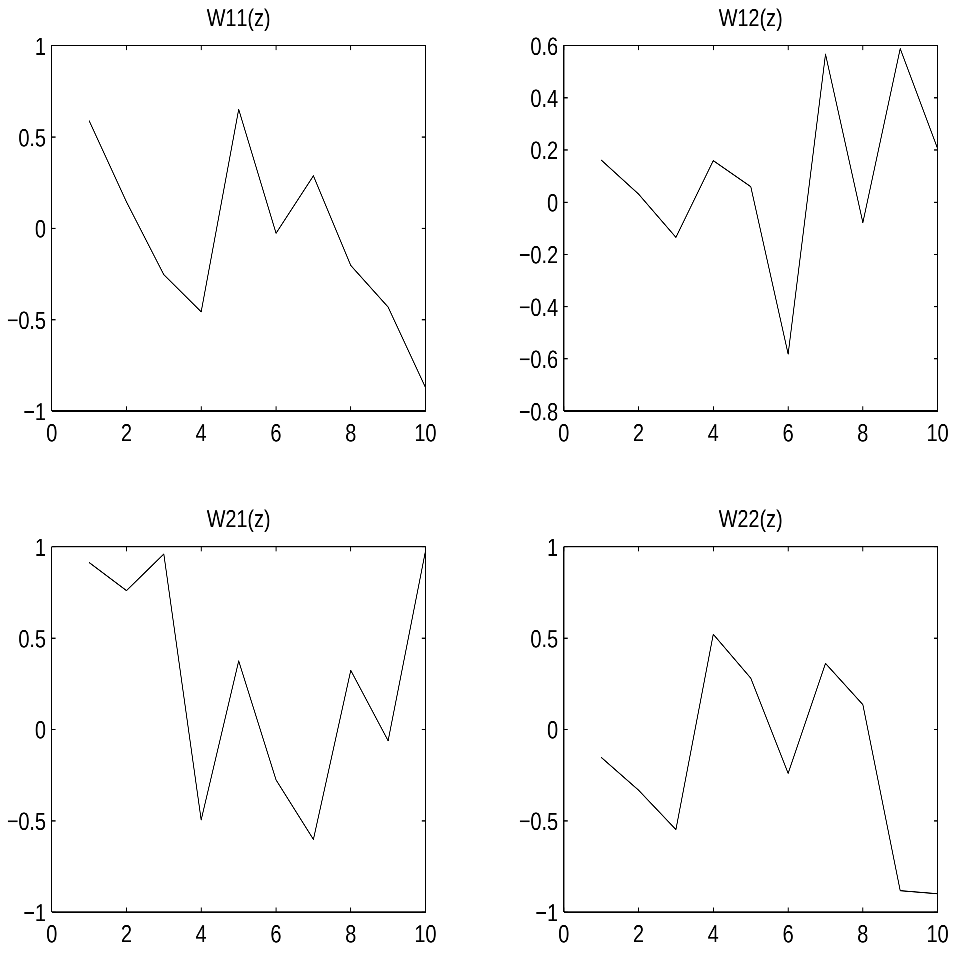

Figure 5.8: Impulse response of demixing filter

it an under-complete situation; when there are more microphones that sources, we call it an over-complete situation.

Perfect separation in the under-complete case is impossible. In this case, the separation algorithm will put the two(or more) most similar sources as one. How to determine the similarity is algorithm dependent. In the Kuicnet method, it means the sources have the closest kurtosis; in the informax method, it means the sources have the closest distribution function.

In the over-complete case there is more information available and thus better separation results can be expected if the number of sources is known and the algorithm is adapted to this information [19]. In addition, operations can be added to remove noise, thus producing a clearer separation result.

In this project, the case of four sensors and two signal sources is studied. The output contains an estimation of the two original signals and of two channels of noise. It is interesting to realize that the two channels of noise have about the same intensity as the estimated source signals.

5.5 Conclusion and future work

In the work I presented here, all current major approaches for BSS/BSD problem are viewed and compared and a real-time implementation of sub-band natural gradient method is provided which works well for real-world recorded voice and music signals. By experimenting with different sensors and source settings, we learned more about how sound is transmitted in a closed room, how sounds are mixed and convolved and how to measure the performance of separation.

Several problem still needs further work to achieve better results:

1. Denoising. The real-world signal contains noise from many sources, for ex-

ample the noise coming from the microphone, and the motor and fan on the robot. The noise is significant in the recorded mixture especially when the robot is moving. Although some wavelet denoising methods are already applied in the algorithm, the noise is not removed completely. Other methods must be considered if a clearer output is desired.

2. Speed of computation. For the instantaneous mixing model, the slave DSP can give an improved demixing matrix for every input and when the robot is moving or the mixing matrix has changed for some other reason, the output can follow the change. For the deconvolution model, the learning process needs more time and cannot follow the change quickly enough. More works need to be done to increase the speed of the algorithm, by either distributing the computational load on the DSP chips more evenly, rewriting more code in assembly, or by employing more advanced algorithms to do the wavelet filtering job.

3. Signal extraction. It would be ideal if the algorithm could extract certain kinds of signals, for example the voice of one person, from a mixture. This requires the use of the signal characteristics in a more thorough way than is currently done.

4. Sound localization. Currently, the sound separation algorithm can not be combined with a localization algorithm because the phase information, which is essential to localization, is damaged in the process of separation. It will be a interesting topic to try combine the two problems together so that the separation can provide more information for localization or sound tracking.

On the theoretical side of the BSS/BSD problem, many problems remain open, such as choice of the best non-linear function, convergence and stability analysis, and adapting the algorithm to different sources and sensor settings. These will attract more more interest form researchers.

We hope to have provided a good review of the BSS/BSD problem and an implementation that provides robust results so that future workers in this field can build on our work and achieve even better results.

Appendix A

Software Structure

A.1 Connection between Coreco board and NT Host

The host program have two major part, executable netsrv.exe and dynamic link library dspsrv.dll. The library dspsrv.dll handles all the requests from DSP board. Some important functions it provides are: CreateCorecoServer(). Coreco Server is a program that reside on NT. Figure A.1 describes how the host program netsrv.exe interact with the code loaded on DSP board natsep.out.

Netsrv.exe call dspsrv.dll behind the scene for all the message handling during run time.

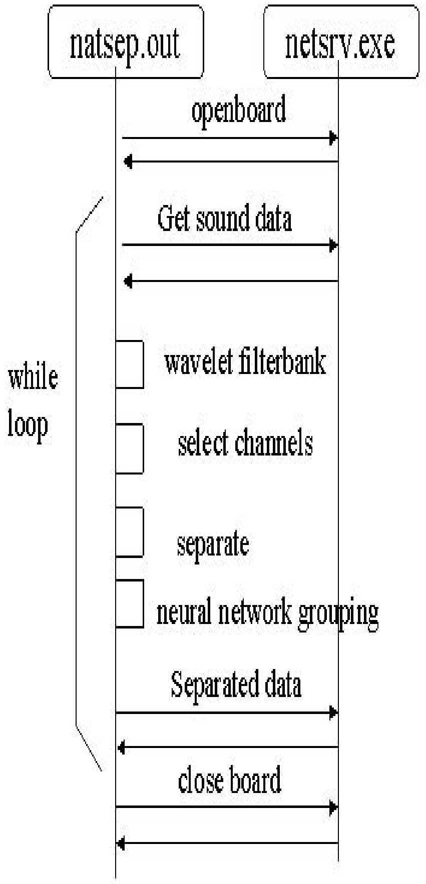

Figure A.1: Host program and server program

A.2 Sub-band ICA algorithm

A.2.1 Data structure

To Implement wavelet filter bank, data structure for filterbank and circular buffer are defined as follows:

1. stereo sound data structure

   typedef struct {

   WAVE_T ch[2];

   } WAVE_STEREO;

   The one sample of sound signal is stored as two 16 bits integer. In each transaction a block of 512 sound signal samples are transferred from robot to NT host as an array of WAVE_STEREO structure.

   Filter structure The information about filter bank is stored in a structure called

2. FILTER_HEAD.

   This structure contains the information about filter bank. There are four fields in this structure. f_type is the enumeration of filter types; depth means the depth of wavelet filter bank tree structure; order means the order of the tree structure, in this project, the value is always 2; nDwords mean the length of filter coefficient vector.

   typedef struct {

UINT32 f_type;

UINT32 depth;

UINT32 order;

UINT32 nDwords;

} FILTER_HEAD;

3. Structure COEF

   COEF is the filter coefficient structure. It's have two fields, pointing to the high pass filter coefficient array and low pass filter coefficient array separately.
   typedef struct {

   COEF_T *H,*L;

   } COEF;

4. Filter Bank

   Data structure for filter bank use both structure FILTER_HEAD and COEF.
   typedef struct {

   FILTER_HEAD *hdr;
   COEF *coef;

   } COEF_BANK;

5. Circular buffer

   Circular buffer is a very important idea of implementing digital filter. In this project, to improve performance, we try to implement filtering and down sampling in one step. Thus the circular buffer is different than standard.

```
typedef struct {
    CIRCULAR_T *ptr, *top, *bottom;
    UINT32 block_leng;
} CIRCULAR;
```

The structure CIRCULAR allocates a whole trunk of memory that can be divided into several blocks each has equal length of block_leng. When we go through the tree structure of wavelet filter-bank, on each level, the original circular buffer is divided into two block contained down sampled data from high pass filter and low pass filter separately. Figure A.2 shows the structure of the sub-band ICA method.

6. Matrix

   To implement ICA algorithm, matrix structure is essential. The structure of matrix is defined as follows

```
typedef struct {
int row_order;
int colunm_order;
float** data;
}mymatrix;
```

   Matrix multiplication, addition, inverse, copy and matrix norm functions are also defined based on this matrix structure.

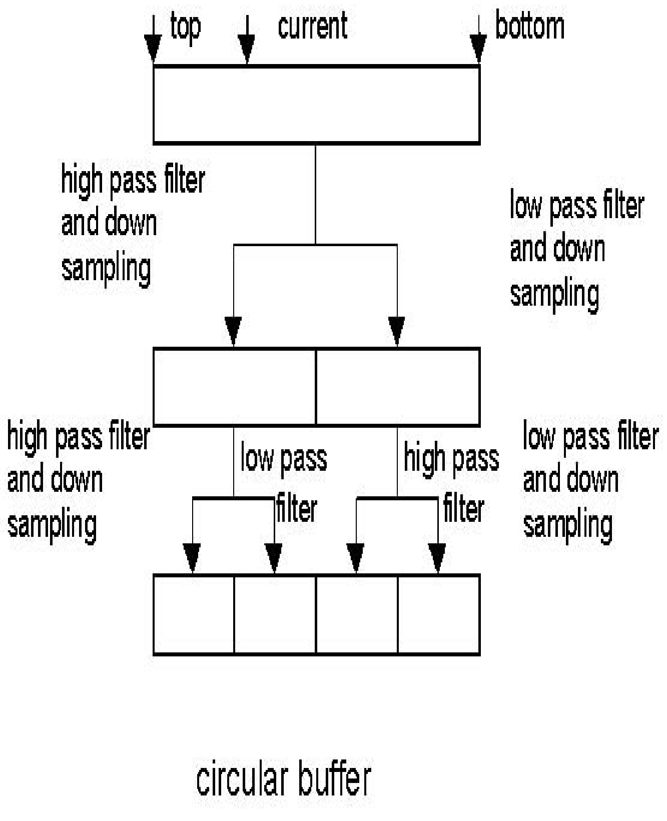
top
current
bottom
high pass filter and down sampling
low pass filter and down sampling
high pass filter and down sampling
low pass filter
high pass filter
low pass filter and down sampling
circular buffer

Figure A.2: Circular buffer and wavelet filter bank

A.2.2 Algorithm

The structure of sub-band ICA algorithm is described in Chapter 3 of thesis. Here we will explain how to implement the algorithm on two DSP board. The graph below describes how two DSP board, the master DSP and slave DSP interact with each other. Figure A.3 shows the structure of the sub-band ICA method.

A.3 File Structure

Main.c: initialize memory, call function *openboard()* to set the connection between Coreco board and NT machine. Then the program goes into an infinite loop. Inside the loop, stereo sound data is read, and function *app()* is called to apply wavelet filtering on the data. For the result, call *sort()* according to average power and than call ICA algorithm to separate. Dbbank.c: contains the wavelet filter bank function *app()*. Firfilter.c: contains all the functions related to filtering calculation which will be used by *app()*. Cluster.c: contains the neural network clustering algorithm. Matrix.c: contains all the functions related to matrix structure and matrix operations. Ica.c: contains the ICA learning algorithm. Code with detailed commented is available under project directory */:/department/isr/labs/isl/Projects/subica/C_code/comment*

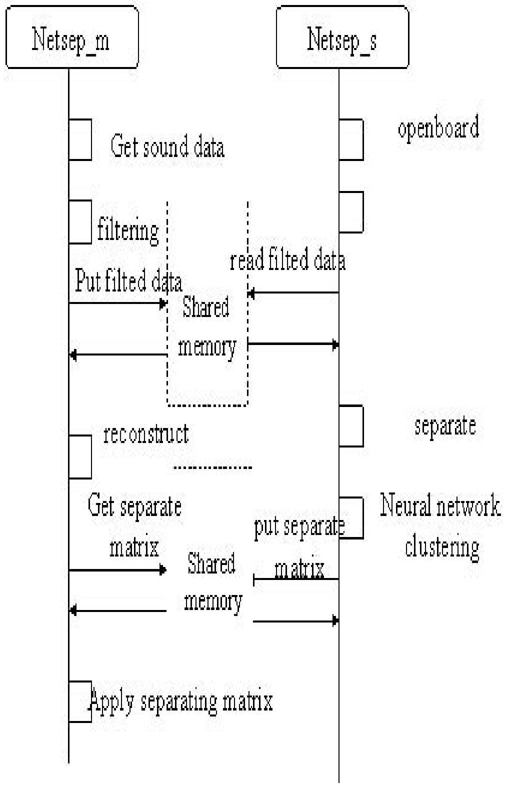

Figure A.3: Master DSP program and slave DSP program

Appendix B

Project Directory

All the source code, data file and pictures mentioned in the thesis is store under directory /:/department/isr/labs/isl/Projects/yumao.

Below is a brief description of directory structure and content inside. All the bold character means directory names.

C_code:

Comment:

the commented code to illustrate the basic idea of implementing sub-band ICA on one DSP board.

Working_ica:

Netsep_m: master DSP code

Netsep_s: slave DSP code

Load the two programs into two DSP for a sub-band ICA implementation The program for master DSP read data, sub-band filtering and apply separating matrix on data. The program for slave DSP applies ICA algorithm on filtered data to get the separating matrix.

Working_dec:

Netsep_m: master DSP code

Netsep_s: slave DSP code

Load the two programs into two DSP boards for a sub-band deconvolution implementation. The program for master DSP read data, sub-band filtering and apply deconvolution filter on data. The program for slave DSP applies deconvolution algorithm on filtered data to get the deconvolution filter.

Coreco_code:

Netsrv.exe and dspsrv.dll: the NT host program

Data:

This sub-directory stores experimental result. All the sound data is stored as binary file. The MATLAB file datareader.m provide an example of reading the stereo sound data and make it acceptable for MATLAB sound function. Note that for NT and Unix system the file format has a little difference. Pictures used in the thesis are stored in postscript format.

Playout:

An executable to play the output sound signal real time. The program use socket to get data from host program, then it write the data directly in to buffer of sound card to enable a smooth sound playing.

BIBLIOGRAPHY

[1] S. Amari. Nature gradient works efficiently in learning. *Neural Computation*, Vol. 10:251–276, 1998.

[2] S. Amari, T.P. Chen, and A. Chichocki. Stability analysis of adaptive blind source separation. *Neural Networks*, Vol. 10, No. 8:1345–1351, 1997.

[3] S. Amari, T.P. Chen, and A. Chichocki. Nonholonomic orthogonal learning algorithms for blind source separation. *Neural Computation*, Vol. 12:1463–1484, 2000.

[4] S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. In *Advances in Neural Information Processing System 8*, pages 757–763. MIT Press, 1996.

[5] S. Amari, S. C. Douglas, A. Cichocki, and H. H. Yang. Multichannel blind deconvolution and equalization using the natural gradient. In *IEEE International Workshop on Wireless Communication, Paris 1997*, pages 101–104, 1997.

[6] H. Attias and C. E. Schreiner. Blind source separation and deconvolution: The dynamic component analysis algorithm. *Neural Computation*, Vol. 10:1373–1424, 1998.

[7] J.-F. Cardoso. On the stability of source separation algorithms. *Procedings of the IEEE*, vol. 86, No. 10, 1998.

[8] R. R. Coifman and M. V. Wickerhauser. Entropy-based algorithms for best basis selection. *IEEE Transactions on Information Theory*, Vol. 38, No. 2, 1992.

[9] T. Cover and J. Thomas, editors. *Information Theory*. John Wiley and Sons Inc., 1991.

[10] D. L. Donoho. De-noising by soft-thresholding. *IEEE Transactions on Information Theory*, Vol. 41, No. 3:613–627, 1995.

[11] S. C. Douglas and S. Haykin. On the relationship between blind deconvolution and blind source separation. In *Proc. 31st Asilomar Conf. Signals, Syste., Comput., Pacific Grove, CA*, volume Vol. 2, pages 1591–1595, 1997.

[12] S. C. Douglas and S.-Y. Kung. Kuicnet algorithms for blind deconvolution. In *Proc. IEEE Workshop on Statistical Signal Array Processing, Portland, OR.*, pages 3–12, 1998.

[13] B. Gold and N. Morgan, editors. *Speech and Audio Signal Processing: Processing and Perception of Speech and Music.* John Wiley and Sons Inc., 2000.

[14] C. Jutten and J. Herault. Blind separation of sources, part i: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 1-10:24, 1991.

[15] B.U. Koehler, T-W. Lee, and R. Orglmeister. Improving the performance of infomax using statistical signal processing techniques. In *Proceedings International Conference on Artificial Neural Networks*, pages 535–540, 1997.

[16] D. MacKay. Maximum likelihood and covariant algorithms for independent component analysis, 1996.

[17] H.-M. Park, H.-Y. Jung, T.-W. Lee, and S.-Y. Lee. On subband-based blind signal separation for noisy speech recognition. *Electronic Letters*, Vol. 35(23):2011–2012, 1999.

[18] Y. Qi, P.S. Krishnaprasad, and S. Shamma. The subband- based independent component analysis. In *Proceedings of ICA'2000*, pages 19–22, 2000.

[19] L. Zhang, S. Amari, and A. Cichocki. Natural gradient approach to blind separation of over- and under-complete mixtures. In *Proceedings of the ICA'99 Aussois, France, 1999*, pages 455–460, 1999.

[20] L. Zhang and A. Cichocki. Blind deconvolution/equalization using state-space models. In *Proceedings of the 1998 IEEE Workshop on Neural Networks for Signal Processing (NNSP'98), Cambridge,UK*, pages 123–131, 1998.